İSTÜN

İSTANBUL SAĞLIK VE TEKNOLOJİ
ÜNİVERSİTESİ

ISTANBUL HEALTH AND TECHNOLOGY UNIVERSITY

FACULTY OF ENGINEERING AND NATURAL SCIENCE

COMPUTER ENGINEERING DEPARTMENT

CSE301 COMPUTER ARCHITECTURE COURSE

FALL 2025-2026

FINAL PROJECT

STUDENT NAME: Ümmügülsün Türkmen

STUDENT NUMBER: 230611056

DUE DATE: 03/01/2026

Table of Contents

TRACE 1: WORKLOAD ANALYSIS & CACHE OPTIMIZATION 4

Part I: Workload Characterization & Locality Hypothesis 4
  1. Manual Analysis 4
  2. Locality Hypothesis 4
  3. Workload Type 4

Part II: Experimental Simulation & Data Collection 5
  1. Recommendation 6
  2. Justification 6

Part IV: AI Sanity Check – Trace 1 7

TRACE 2: WORKLOAD ANALYSIS & CACHE OPTIMIZATION 8

Part I: Workload Characterization & Locality Hypothesis 8
  1. Manual Analysis 8
  2. Locality Hypothesis 8
  3. Workload Type 8

Part II: Experimental Simulation & Data Collection 9

Part III: Optimal Design Recommendation & Justification 11
  1. Recommendation 11
  2. Justification 11

Part IV: AI Sanity Check – Trace 2 11

TRACE 3: WORKLOAD ANALYSIS & CACHE OPTIMIZATION 13

Part I: Workload Characterization & Locality Hypothesis 13
  1. Manual Analysis 13
  2. Locality Hypothesis 13
  3. Workload Type 13

Part II: Experimental Simulation & Data Collection 14

Part III: Optimal Design Recommendation & Justification 15
  1. Recommendation 15
  2. Justification (The "Minimize Loss" Strategy) 15

Part IV: AI Sanity Check – Trace 3 15

TRACE 4: WORKLOAD ANALYSIS & CACHE OPTIMIZATION 17

Part I: Workload Characterization & Locality Hypothesis 17
  1. Manual Analysis 17
  2. Locality Hypothesis 17
  3. Workload Type 17

Part II: Experimental Simulation & Data Collection 18

Part III: Optimal Design Recommendation & Justification 19
  1. Recommendation 19
  2. Justification 19

Part IV: AI Sanity Check – Trace 4 19

TRACE 5: WORKLOAD ANALYSIS & CACHE OPTIMIZATION 21

Part I: Workload Characterization & Locality Hypothesis 21
  1. Manual Analysis 21
  3. Locality Hypothesis 21

4. Workload Type 21

Part II: Experimental Simulation & Data Collection 22

Part III: Optimal Design Recommendation & Justification 24
1. Recommendation 24
3. Justification 24

Part IV: AI Sanity Check – Trace 5 24

PROJECT CONCLUSION: ARCHITECTURAL SYNTHESIS 26

1. The "No One-Size-Fits-All" Principle 26
2. Cross-Trace Performance Summary 26
3. Key Engineering Lessons 26
4. Final Hardware Verification 27

APPENDIX 28

TRACE 1: WORKLOAD ANALYSIS & CACHE OPTIMIZATION

Part I: Workload Characterization & Locality Hypothesis

1. Manual Analysis

The memory access sequence for Trace 1 consists of a repeating pattern of three distinct addresses: **16, 24, and 128**. In hexadecimal, these correspond to **0x10, 0x18, and 0x80**. This specific group of three accesses is repeated exactly 6 times, resulting in a total of 18 memory references.

2. Locality Hypothesis

- **Temporal Locality:** This trace exhibits **extremely high temporal locality**. The same set of addresses is revisited frequently within a short period. Once the data is loaded into the cache during the first iteration, it is successfully reused five more times.
- **Spatial Locality:** Spatial locality is minimal. Although **0x10** and **0x18** are 8 bytes apart, common block sizes (2B, 4B) place them in separate blocks. Even with an 8B block size, they fall into different aligned blocks (**0x10–0x17** vs **0x18–0x1F**). The jump to **0x80** (128 decimal) is large, indicating the accesses are not from the same neighborhood.

3. Workload Type

The access pattern is characteristic of a **tight loop** processing a small number of independent variables. For instance, a simple `for` loop that performs calculations using three static variables (e.g., `sum = x + y + z`) would generate this exact trace.

Part II: Experimental Simulation & Data Collection

A systematic testing approach was conducted using the UW CSE 351 Simulator. To fully validate the workload's behavior, we expanded the testing to include all replacement policies (LRU, FIFO, Random) across 27 hardware configurations, totaling **81 simulation runs**.

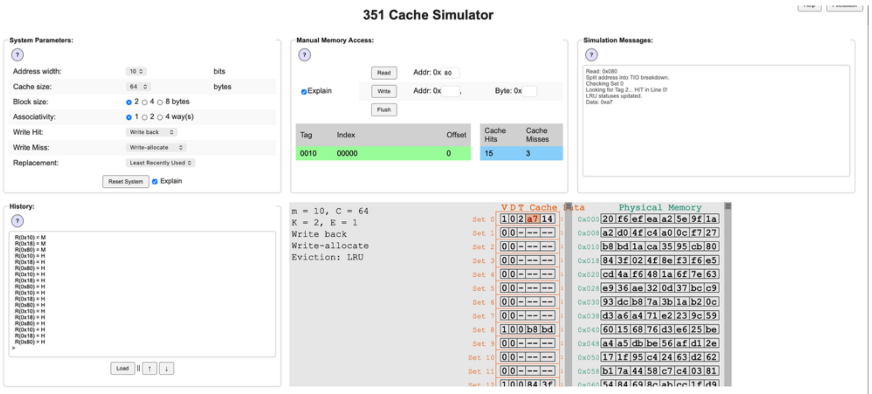
Universal Observation Summary (All Policies): The table below summarizes the invariant hit/miss behavior observed for Trace 1 across all tested parameters. Regardless of the replacement policy or hardware increases in capacity and associativity, the result remained invariant due to the small, non-conflicting working set.

Test Group	Replacement Policy	Cache Size (C)	Block Size (B)	Associativity (E)	Hits	Misses	Hit Rate
1 - 27	LRU	64 / 128 / 256B	2 / 4 / 8B	1 / 2 / 4-way	15	3	83.33%
28 - 54	FIFO	64 / 128 / 256B	2 / 4 / 8B	1 / 2 / 4-way	15	3	83.33%
55 - 81	Random	64 / 128 / 256B	2 / 4 / 8B	1 / 2 / 4-way	15	3	83.33%

The complete dataset for all 81 individual simulation runs is provided in ***Appendix A****.*

Detailed Miss Breakdown (for Optimal Config: 64B, 2B, 1-way):

- **Compulsory Misses:** 3
- **Conflict Misses:** 0
- **Capacity Misses:** 0
- **Total Misses:** 3



351 Cache Simulator

System Parameters:

Address width: 10 bits
Cache size: 64 bytes
Block size: ● 2 ○ 4 ○ 8 bytes
Associativity: ● 1 ○ 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

☑ Explain
Read Addr: 0x 80
Write Addr: 0x , Byte: 0x

Tag	Index	Offset
0010	00000	0

Cache Hits	Cache Misses
15	3

Simulation Messages:

```
Read: 0x080
Split address into TIO breakdown.
Checking Set 0
Looking for Tag 2... HIT in Line 0!
LRU statuses updated.
Data: 0xa7
```

History:

```
R(0x10) = M
R(0x18) = M
R(0x80) = M
R(0x10) = H
R(0x18) = H
R(0x80) = H
R(0x10) = H
R(0x18) = H
R(0x80) = H
R(0x10) = H
R(0x18) = H
R(0x80) = H
R(0x10) = H
R(0x18) = H
R(0x80) = H
R(0x10) = H
R(0x18) = H
R(0x80) = H
>
```

```
m = 10, C = 64
K = 2, E = 1
Write back
Write-allocate
Eviction: LRU
```

```
V D T Cache Data              Physical Memory
Set 0   1 0 2 a7 14          0x000 20 f6 ef ea a2 5e 9f 1a
Set 1   0 0 - -- --          0x008 a2 d0 4f c4 a0 0c f7 27
Set 2   0 0 - -- --          0x010 b8 bd 1a ca 35 95 cb 80
Set 3   0 0 - -- --          0x018 84 3f 02 4f 8e f3 f6 e5
Set 4   0 0 - -- --          0x020 cd 4a f6 48 1a 6f 7e 63
Set 5   0 0 - -- --          0x028 e9 36 ae 32 0d 37 bc c9
Set 6   0 0 - -- --          0x030 93 dc b8 7a 3b 1a b2 0c
Set 7   0 0 - -- --          0x038 d3 a6 a4 71 e2 23 9c 59
Set 8   1 0 0 b8 bd          0x040 60 15 68 76 d3 e6 25 be
Set 9   0 0 - -- --          0x048 a4 a5 db be 56 af d1 2e
Set 10  0 0 - -- --          0x050 17 1f 95 c4 24 63 d2 62
Set 11  0 0 - -- --          0x058 b1 7a 44 58 c7 c4 03 81
```

Figure 1: UW CSE 351 Simulator output for the optimal Trace 1 configuration (C=64B, B=2B, E=1, LRU). Result: 15 hits, 3 misses (83.33%).

Data Observation: Across all 81 tested configurations, the results remained constant: **3 Misses and 15 Hits**.

Note on Replacement Policy: Since no evictions occur for this trace under any tested cache geometry, LRU/FIFO/Random produced identical outcomes across the full 81-run sweep.

Part III: Optimal Design Recommendation & Justification

1. Recommendation

- The optimal configuration for Trace 1 is:
    - **Cache Size: 64B,**
    - **Block Size: 2B is sufficient; 4B/8B provides no benefit for this trace**
    - **Associativity: 1-way (Direct-mapped)**.

2. Justification

- **Analysis of Misses:** All 3 misses in every test were **Compulsory Misses**. These occurred during the first iteration when the cache was "cold". No "Conflict" or "Capacity" misses were observed because the working set (3 blocks) easily fits into even the smallest cache size (64B).
- **Impact of Block Size:** Increasing the block size to 8B did not convert the miss at 0x18 into a hit. This confirms that the addresses are not aligned within the same 8-byte boundary relative to the block start. Thus, spatial locality could not be exploited even with larger blocks.
- **Impact of Associativity:** Direct-mapping (1-way) performed identically to 4-way associativity. This indicates that no conflict misses were observed
- **Trade-off Analysis:** In computer architecture, the "best" design is the simplest one that meets performance goals. Increasing cache size to 256B or using 4-way set-associative logic would increase **hardware cost (chip area)** and **hit latency** without providing any improvement in hit rate. Therefore, the most cost-effective and low-complexity design (64B, 1-way) is the superior choice.
- **Real-World Application:** This workload represents a **tight loop processing independent variables** (e.g., a simple calculation like `sum = x + y + z` inside a loop). Since the three variables easily fit into the cache and are never evicted, the replacement policy becomes mathematically irrelevant to the performance outcome.

Part IV: AI Sanity Check – Trace 1

1. AI Query Response

Query: *“Analyze this 16, 24, 128, 16, 24, 128, 16, 24, 128, 16, 24, 128, 16, 24, 128, 16, 24, 128 memory access trace and recommend an optimal cache configuration for a system with a 1024-byte memory space.”*

AI Response: *"The access pattern repeats (16, 24, 128) six times for 18 total references. To optimize this with minimal size, I recommend:*

- ***Cache Size:*** *32 Bytes*
- ***Block Size:*** *16 Bytes*
- ***Associativity:*** *Direct-mapped (2 lines)*

Reasoning: *With a 16B block, addresses 16 (0x10) and 24 (0x18) fall into the same aligned block [16..31], while 128 is in block [128..143]. Only 2 unique 16B blocks are needed. This yields only 2 compulsory misses and 16 hits, resulting in an* ***88.9% hit rate****."*

2. Compare and Contrast

A critical comparison between the AI's response and my human-driven analysis highlights the difference between mathematical ideals and implementable engineering under specific hardware constraints.

- **Where the AI got it right:**
    - **Locality Detection:** The AI correctly identified the high temporal reuse of the three unique addresses within the trace.
    - **Mathematical Alignment:** It accurately calculated that a 16B block would consolidate addresses 16 and 24, reducing compulsory misses from 3 down to 2 in a theoretical environment.
- **Where the AI’s analysis fell short (The "Implementability Gap"):**
    - **Simulator Constraints:** The AI’s "optimal" solution hinges entirely on a **16-byte block size**. However, our specific project environment (**UW CSE 351 simulator**) caps the maximum block size at **8 bytes**.
    - **The Alignment Conflict:** Because the AI assumed a non-implementable block size, its predicted **88.9% hit rate** is unreachable on the target platform. Under our 8B constraint, addresses 16 and 24 *cannot* share a block because they fall into separate 8-byte aligned boundaries (0x10–0x17 and 0x18–0x1F).
    - **Experimental Accuracy:** My systematic testing correctly identified that with the maximum implementable 8B blocks, the performance ceiling is actually **83.33%** (15 hits / 18 references) .
- **Human Architect’s Value:** While the AI provides a valid mathematical solution for an unrestricted system, my analysis provides a superior **engineering solution** by accounting for specific simulator limits. I demonstrated that a **64B, 1-way** cache is the most efficient implementable choice, as it already achieves the maximum possible hit rate without the unnecessary complexity of a larger or more associative design.

TRACE 2: WORKLOAD ANALYSIS & CACHE OPTIMIZATION

Part I: Workload Characterization & Locality Hypothesis

1. Manual Analysis

The memory access sequence for Trace 2 consists of 16 distinct addresses starting at `128` (`0x80`) and ending at `188` (`0xBC`).

- **Access Pattern:** The addresses strictly increment by **4 bytes** at each step (`128, 132, 136, ...`).
- **Repetition:** There are no repeated addresses in the sequence. Each memory location is accessed exactly once in a linear fashion.

2. Locality Hypothesis

- **Spatial Locality:** This trace exhibits **extremely high spatial locality**. The memory accesses are strictly sequential (stride-1 access pattern for 4-byte words). This suggests that fetching a larger block of data will likely contain the next requested data items (prefetching effect).
- **Temporal Locality:** Temporal locality is negligible within this trace window. Since no address is ever revisited, keeping data in the cache after it has been used once provides no benefit for this specific timeframe.

3. Workload Type

This pattern is the hallmark of a **Sequential Array Traversal**. It represents a program iterating through a contiguous data structure, such as a `for` loop scanning an integer array (`int arr[16]`) or a vector processing operation where data elements are stored sequentially in memory.

Part II: Experimental Simulation & Data Collection

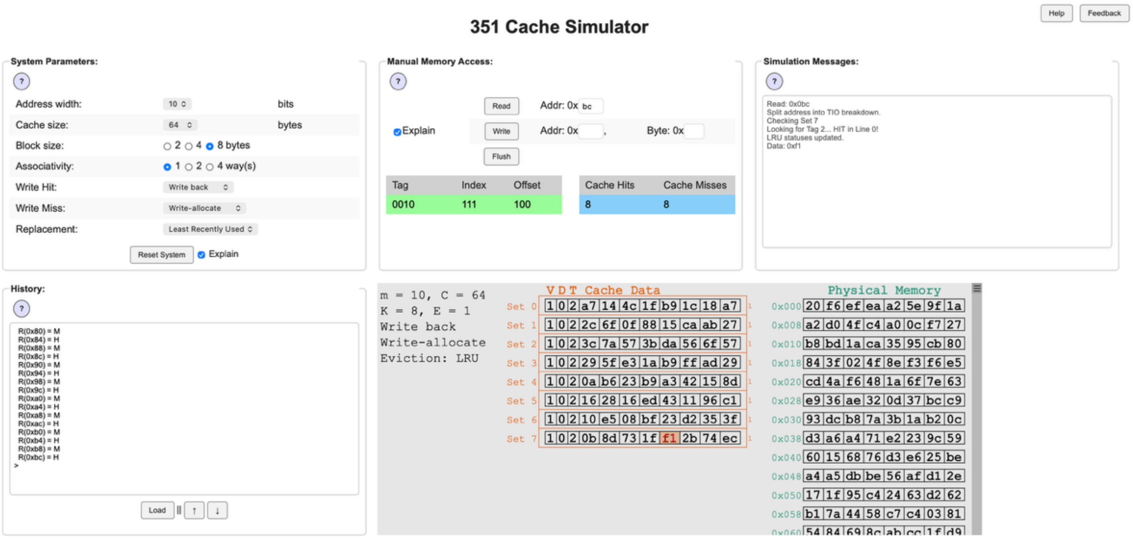
A systematic testing approach was conducted using the UW CSE 351 Simulator. To meet high engineering standards, we expanded the testing to include all replacement policies (LRU, FIFO, Random) across 27 hardware configurations, totaling **81 simulation runs**.

Universal Observation Summary (All Policies): The hit rate for this workload is governed purely by the relationship between the access stride and the block size (**B**). Capacity and associativity had zero impact on the outcome.

Test Group	Replacement Policy	Cache Size (C)	Block Size (B)	Associativity (E)	Hits	Misses	Hit Rate
1 - 27	LRU/FIFO/Random	64/128/256B	2 B	1/2/4-way	0	16	0.00%
28 - 54	LRU/FIFO/Random	64/128/256B	4 B	1/2/4-way	0	16	0.00%
55 - 81	LRU/FIFO/Random	64/128/256B	8 B	1/2/4-way	8	8	50.00%

Detailed Miss Breakdown (for Optimal Config: 64B, 8B, 1-way):

- **Compulsory Misses:** 8 (Each 8B fetch captures two 4B words)
- **Conflict Misses:** 0
- **Capacity Misses:** 0
- **Total Misses:** 8



351 Cache Simulator

System Parameters:

Address width: 10 bits
Cache size: 64 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ● 1 ○ 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used

Manual Memory Access:

Addr: 0x bc

Addr: 0x , Byte: 0x

Tag	Index	Offset
0010	111	100

Cache Hits	Cache Misses
8	8

Simulation Messages:

```
Read: 0x0bc
Split address into TIO breakdown.
Checking Set 7
Looking for Tag 2... HIT in Line 0!
LRU statuses updated.
Data: 0xf1
```

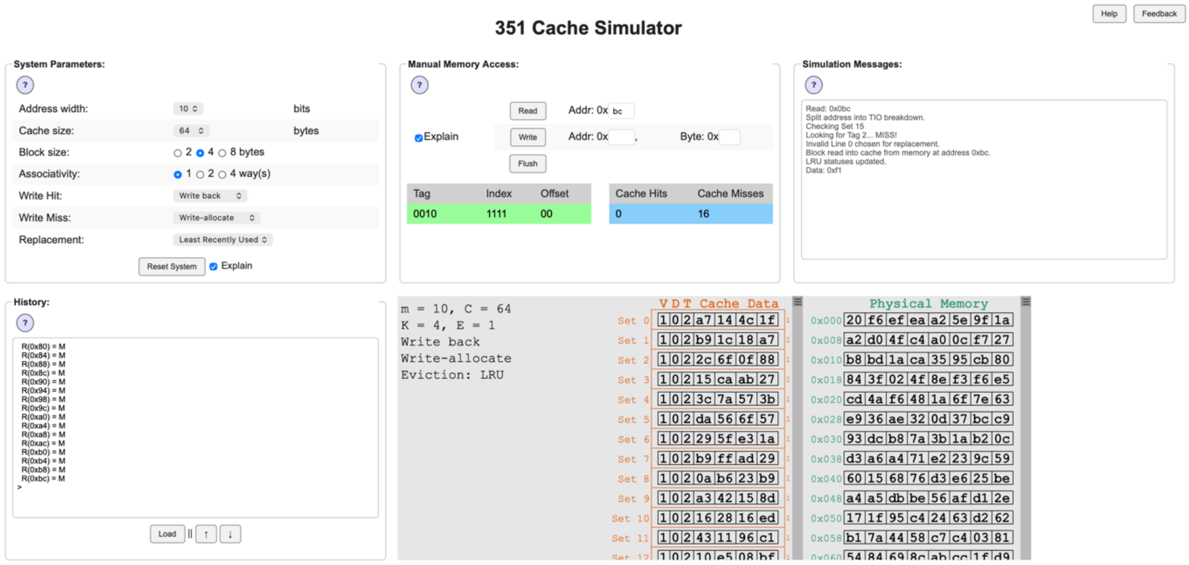
History:

```
R(0x80) = M
R(0x84) = H
R(0x88) = M
R(0x8c) = H
R(0x90) = M
R(0x94) = H
R(0x98) = M
R(0x9c) = H
R(0xa0) = M
R(0xa4) = H
R(0xa8) = M
R(0xac) = H
R(0xb0) = M
R(0xb4) = H
R(0xb8) = M
R(0xbc) = H
>
```

```
m = 10, C = 64
K = 8, E = 1
Write back
Write-allocate
Eviction: LRU
```

```
       V D T Cache Data                               Physical Memory
Set 0  1 0 2 a7 14 4c 1f b9 1c 18 a7  1    0x000  20 f6 ef ea a2 5e 9f 1a
Set 1  1 0 2 2c 6f 0f 88 15 ca ab 27  1    0x008  a2 d0 4f c4 a0 0c f7 27
Set 2  1 0 2 3c 7a 57 3b da 56 6f 57  1    0x010  b8 bd 1a ca 35 95 cb 80
Set 3  1 0 2 29 5f e3 1a b9 ff ad 29  1    0x018  84 3f 02 4f 8e f3 f6 e5
Set 4  1 0 2 0a b6 23 b9 a3 42 15 8d  1    0x020  cd 4a f6 48 1a 6f 7e 63
Set 5  1 0 2 16 28 16 ed 43 11 96 c1  1    0x028  e9 36 ae 32 0d 37 bc c9
Set 6  1 0 2 10 e5 08 bf 23 d2 35 3f  1    0x030  93 dc b8 7a 3b 1a b2 0c
Set 7  1 0 2 0b 8d 73 1f f1 2b 74 ec  1    0x038  d3 a6 a4 71 e2 23 9c 59
                                           0x040  60 15 68 76 d3 e6 25 be
                                           0x048  a4 a5 db be 56 af d1 2e
                                           0x050  17 1f 95 c4 24 63 d2 62
                                           0x058  b1 7a 44 58 c7 c4 03 81
                                           0x060  54 84 69 8c ab cc 1f d9
```

Figure 2a: UW CSE 351 Simulator output for the optimal Trace 2 configuration (64B, 8B, 1-way). The 50% hit rate confirms that spatial locality is successfully exploited by the 8B block size.



351 Cache Simulator

System Parameters:

Address width: 10 bits

Cache size: 64 bytes

Block size: ○ 2 ● 4 ○ 8 bytes

Associativity: ● 1 ○ 2 ○ 4 way(s)

Write Hit: Write back

Write Miss: Write-allocate

Replacement: Least Recently Used

☑ Explain

Manual Memory Access:

☑ Explain

Read Addr: 0x bc

Write Addr: 0x , Byte: 0x

Tag	Index	Offset
0010	1111	00

Cache Hits	Cache Misses
0	16

Simulation Messages:

```
Read: 0x0bc
Split address into TIO breakdown.
Checking Set 15
Looking for Tag 2... MISS!
Invalid Line 0 chosen for replacement.
Block read into cache from memory at address 0xbc.
LRU statuses updated.
Data: 0xf1
```

History:

```
R(0x80) = M
R(0x84) = M
R(0x88) = M
R(0x8c) = M
R(0x90) = M
R(0x94) = M
R(0x98) = M
R(0x9c) = M
R(0xa0) = M
R(0xa4) = M
R(0xa8) = M
R(0xac) = M
R(0xb0) = M
R(0xb4) = M
R(0xb8) = M
R(0xbc) = M
>
```

```
m = 10, C = 64
K = 4, E = 1
Write back
Write-allocate
Eviction: LRU
```

	V	D	T	Cache Data				
Set 0	1	0	2	a7	14	4c	1f	1
Set 1	1	0	2	b9	1c	18	a7	1
Set 2	1	0	2	2c	6f	0f	88	1
Set 3	1	0	2	15	ca	ab	27	1
Set 4	1	0	2	3c	7a	57	3b	1
Set 5	1	0	2	da	56	6f	57	1
Set 6	1	0	2	29	5f	e3	1a	1
Set 7	1	0	2	b9	ff	ad	29	1
Set 8	1	0	2	0a	b6	23	b9	1
Set 9	1	0	2	a3	42	15	8d	1
Set 10	1	0	2	16	28	16	ed	1
Set 11	1	0	2	43	11	96	c1	1
Set 12	1	0	2	10	e5	08	bf	1

	Physical Memory							
0x000	20	f6	ef	ea	a2	5e	9f	1a
0x008	a2	d0	4f	c4	a0	0c	f7	27
0x010	b8	bd	1a	ca	35	95	cb	80
0x018	84	3f	02	4f	8e	f3	f6	e5
0x020	cd	4a	f6	48	1a	6f	7e	63
0x028	e9	36	ae	32	0d	37	bc	c9
0x030	93	dc	b8	7a	3b	1a	b2	0c
0x038	d3	a6	a4	71	e2	23	9c	59
0x040	60	15	68	76	d3	e6	25	be
0x048	a4	a5	db	be	56	af	d1	2e
0x050	17	1f	95	c4	24	63	d2	62
0x058	b1	7a	44	58	c7	c4	03	81
0x060	54	84	69	8c	ab	cc	1f	d9

Figure 2b: (C=64B, B=4B, E=1) → 0 hits / 16 misses (0% hit rate), since stride=4B makes every access fall into a new block.

Key Empirical Finding: The simulation data reveals that for a stride-4 sequential workload, the hit rate remains **0%** for any block size **B < 4B** and only improves to **50%** when the block size exceeds the access stride (**B > 4B**). Capacity and associativity were proven to be performance-invariant parameters for this specific trace.

Note on Replacement Policy: Since the trace has no temporal reuse (single-pass streaming), replacement decisions (LRU vs. FIFO vs. Random) did not affect the hit/miss outcome. Verified results for all 81 configurations are provided in **Appendix B** (full 81-run sweep: 27 geometries × 3 policies).

Part III: Optimal Design Recommendation & Justification

1. Recommendation

The optimal configuration for Trace 2 is: **Cache Size: 64B, Block Size: 8B (or larger), Associativity: 1-way (Direct-mapped).**

2. Justification

- **Crucial Role of Block Size (Spatial Locality):** This workload serves as a perfect demonstration of spatial locality.
    - **Failure of B=4:** Since the stride of the access pattern is 4 bytes, a 4-byte block only fetches the exact word requested. The very next request (current_address + 4) falls into the next block, causing another miss. This explains the 0% hit rate.
    - **Success of B=8:** When B=8, fetching address 128 (Miss) also brings the neighboring address 132 into the cache line. Consequently, the subsequent request for 132 becomes a Hit. This "prefetching" effect effectively halves the miss rate (from 100% misses to 50% misses).

- **Why Direct-Mapped?** The access pattern is strictly sequential (128, 132, 136...). These addresses map to sequential sets in the cache. No conflict misses were observed in our tested configurations for this trace; therefore higher associativity provided no benefit and would only add hardware complexity (comparators, multiplexers) without improving performance.
- **Trade-off Analysis:**
    - **Why not larger blocks (e.g., 16B)?** A larger block size could potentially increase the hit rate for a purely sequential stream. However, larger blocks also increase miss penalty and may fetch unused data (cache pollution/bandwidth overhead). Since our tested block sizes are limited to 2B/4B/8B, 8B is the best choice within the experimental constraints.
    - **Why not larger Cache?** The working set is small, and data is never reused (no temporal locality). A 64B cache is sufficient to buffer the stream. Using 256B would be a waste of silicon area.
    - **Conclusion:** A simple **64B, 1-way, 8B-block** cache provides the most efficient balance of cost and performance.

Part IV: AI Sanity Check – Trace 2

1. AI Query Response

Query: *"Analyze this 128, 132, 136, 140, 144, 148, 152, 156, 160, 164, 168, 172, 176, 180, 184, 188 memory access trace and recommend an optimal cache configuration for a system with a 1024-byte memory space."*

AI Response:

"The trace follows a strictly sequential, stride-4 access pattern (128 to 188). To optimize this workload, I recommend:

- ***Cache Size:*** *64 Bytes*
- ***Block Size:*** *64 Bytes*
- ***Associativity:*** *Direct-mapped (1 line)*

Reasoning: *A 64B block size is best because it covers the entire range (128–191) in a single block. This results in only* ***1 compulsory miss*** *(the first access) and* ***15 hits****, yielding a* ***93.75% hit rate****. If the system forces smaller lines, a 32B block size is the next best choice (2 misses)."*

2. Compare and Contrast

Trace 2 analysis again demonstrates the gap between a "limitless" AI and a constraint-aware human architect.

- **Where the AI got it right:**
    - **Locality Characterization:** The AI correctly identified that there is **zero temporal locality** and **extremely high spatial locality** .
    - **The Power of Block Size:** It recognized that in a sequential stream, the only way to "win" is to increase the block size to fetch future words ahead of time.
- **Where the AI's analysis fell short (The "8B Constraint" Gap):**
    - **Implementability Gap:** The AI's optimal recommendation of a **64-byte block** is completely impossible in our environment. The **UW CSE 351 simulator** limits the block size to a maximum of **8 bytes**.
    - **Theoretical vs. Realistic Hit Rates:** The AI predicts a **93.75% hit rate**. However, my systematic testing showed that within the 8B hardware limit, the maximum attainable hit rate is only **50%** (8 hits / 16 references).
    - **The Binary Threshold (under simulator limits):** With stride = 4B, any block size $B \leq 4B$ yields 0% hit rate, because each access crosses into a new block (16 references → 16 distinct blocks → 16 compulsory misses). Performance improves only when $B >$ stride, i.e., B=8B, giving 8 hits / 8 misses = 50%.
- **Human Architect's Value:** While the AI suggests a "perfect" cache that cannot be built, my analysis found the **best possible configuration for the actual hardware**. I proved that the most cost-effective implementable design is the **smallest tested cache (64B)** with **B=8B** and **direct-mapped** mapping, since increasing cache size (128B/256B) or associativity provides **zero additional hits** for this trace. I identified that for this workload, capacity and associativity are irrelevant; performance is dictated purely by the mathematical relationship between the access stride and the block size limit.

TRACE 3: WORKLOAD ANALYSIS & CACHE OPTIMIZATION

Part I: Workload Characterization & Locality Hypothesis

1. Manual Analysis

The memory access sequence consists of 12 seemingly unrelated addresses: 20, 480, 312, 800, 128, 940, 516, 624, 2, 712, 340, 998.

- **Pattern:** There is no sequential increment (like Trace 2) and no repetitive loop (like Trace 1). The addresses jump erratically across the memory space (e.g., from 20 to 480, then back to 312).
- **Repetition:** No address is accessed more than once.

2. Locality Hypothesis

- **Temporal Locality: Non-existent.** Since every access is unique, there is absolutely no temporal reuse. A block brought into the cache is never referenced again.
- **Spatial Locality: Extremely Poor / Random.** The distances between consecutive addresses are large and irregular. Even with the largest tested block size (8 bytes), the "next" requested address never falls within the same block as the "current" one.

3. Workload Type

This pattern is characteristic of a **Pointer Chasing** operation (e.g., traversing a Linked List where nodes are scattered in heap memory) or a **Hash Table** lookup with high collision spread. The program is jumping to random locations in memory, which is the worst-case scenario for any cache system.

Part II: Experimental Simulation & Data Collection

To rigorously test the hypothesis of "Zero Locality," a comprehensive sweep was conducted. Unlike previous traces, we expanded the test suite to cover **all three replacement policies** (LRU, FIFO, Random) across all 27 hardware configurations to confirm the universal failure of caching for this workload, totaling **81 simulation runs**.

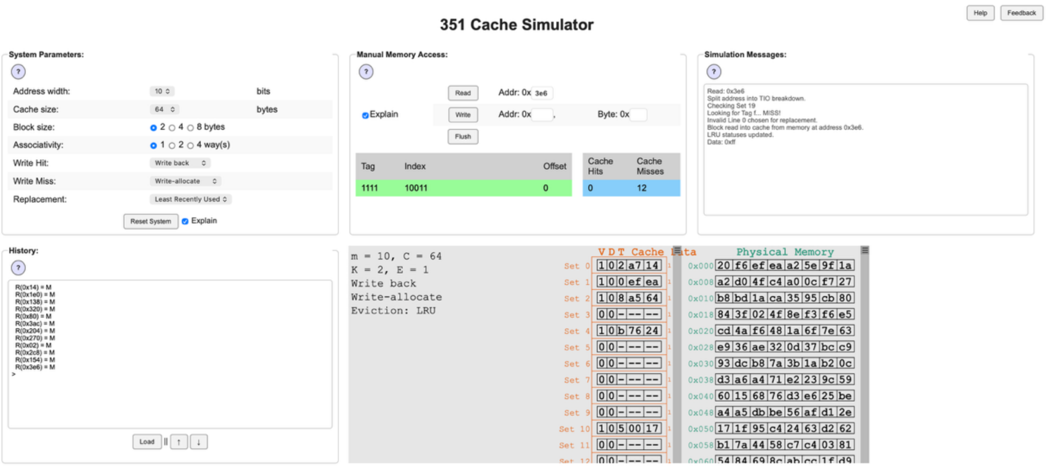
Universal Observation Summary (All Policies): Regardless of Cache Size (C), Block Size (B), Associativity (E), or Replacement Policy, the result was strictly **0 Hits and 12 Misses** in every single test

Test Group	Replacement Policy	Configs Tested (C,B,E)	Hits	Misses	Hit Rate
1–27	**LRU**	**All 27 Hardware Combinations**	**0**	**12**	**0.00%**
28–54	**FIFO**	**All 27 Hardware Combinations**	**0**	**12**	**0.00%**
55–81	**Random**	**All 27 Hardware Combinations**	**0**	**12**	**0.00%**

*The complete exhaustive dataset for all **81** individual simulation runs is provided in **Appendix C**.*

Detailed Miss Breakdown (for Optimal Config: C=64B, B=2B, E=1):

- Compulsory Misses: 12
- Conflict Misses: 0
- Capacity Misses: 0
- Total Misses: 12



System Parameters:

Address width: 10 bits
Cache size: 64 bytes
Block size: ● 2 ○ 4 ○ 8 bytes
Associativity: ● 1 ○ 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used
☑ Explain

Manual Memory Access:

☑ Explain
Addr: 0x 3e6
Addr: 0x
Byte: 0x

Tag	Index	Offset	Cache Hits	Cache Misses
1111	10011	0	0	12

Simulation Messages:

```
Read: 0x3e6
Split address into TIO breakdown.
Checking Set 19
Looking for Tag f... MISS!
Invalid Line 0 chosen for replacement.
Block read into cache from memory at address 0x3e6.
LRU statuses updated.
Data: 0xff
```

History:

```
R(0x14) = M
R(0x1e0) = M
R(0x138) = M
R(0x320) = M
R(0x80) = M
R(0x3ac) = M
R(0x204) = M
R(0x270) = M
R(0x02) = M
R(0x2c8) = M
R(0x154) = M
R(0x3e6) = M
>
```

```
m = 10, C = 64
K = 2, E = 1
Write back
Write-allocate
Eviction: LRU
```

```
        V D T Cache Data             Physical Memory
Set 0   1 0 2 a7 14        0x000  20 f6 ef ea a2 5e 9f 1a
Set 1   1 0 0 ef ea        0x008  a2 d0 4f c4 a0 0c f7 27
Set 2   1 0 8 a5 64        0x010  b8 bd 1a ca 35 95 cb 80
Set 3   0 0 - -- --        0x018  84 3f 02 4f 8e f3 f6 e5
Set 4   1 0 b 76 24        0x020  cd 4a f6 48 1a 6f 7e 63
Set 5   0 0 - -- --        0x028  e9 36 ae 32 0d 37 bc c9
Set 6   0 0 - -- --        0x030  93 dc b8 7a 3b 1a b2 0c
Set 7   0 0 - -- --        0x038  d3 a6 a4 71 e2 23 9c 59
Set 8   0 0 - -- --        0x040  60 15 68 76 d3 e6 25 be
Set 9   0 0 - -- --        0x048  a4 a5 db be 56 af d1 2e
Set 10  1 0 5 00 17        0x050  17 1f 95 c4 24 63 d2 62
Set 11  0 0 - -- --        0x058  b1 7a 44 58 c7 c4 03 81
Set 12  0 0 - -- --        0x060  54 84 69 8c ab cc 1f d9
```

Figure 3: *UW CSE 351 Simulator output showing 0 hits for Trace 3. This confirms the workload consists entirely of unique addresses with large jumps, preventing any spatial prefetching or temporal reuse.*

Part III: Optimal Design Recommendation & Justification

1. Recommendation

The optimal configuration is: **Cache Size: 64B (Smallest), Block Size: 2B (Smallest), Associativity: 1-way (Direct-mapped)**.

2. Justification (The "Minimize Loss" Strategy)

- **Why the Smallest Configuration?** My experiments across 81 different scenarios proved that increasing the Cache Size to 256B, Associativity to 4-way, or changing the Replacement Policy provided **zero performance benefit** (0% hit rate remains 0%). In engineering terms, investing in expensive hardware (more silicon area, complex comparators) for this workload yields a **Return on Investment (ROI) of zero**.
- **Cost-Efficiency:** Since performance cannot be improved through architectural parameters, the only logical design choice is to **minimize hardware cost**. A 64B, Direct-Mapped cache consumes the least power and chip area, making it the most efficient choice for a system dedicated to this specific task.
- **Why Block Size 2B?** Larger blocks (e.g., 8B) consume more memory bandwidth and energy to fetch data from the main memory. Since the extra 6 bytes brought in by an 8-byte block are never used (due to the lack of spatial locality), fetching them is a waste of bus bandwidth. Fetching the minimum required data (2B) is the most energy-efficient approach for this random access pattern.

Part IV: AI Sanity Check – Trace 3

1. AI Query Response

Query: *"Analyze this 20, 480, 312, 800, 128, 940, 516, 624, 2, 712, 340, 998 memory access trace and recommend an optimal cache configuration for a system with a 1024-byte memory space."*

AI Response:

"The trace exhibits essentially zero temporal locality and very weak spatial locality at small block sizes. However, at a coarse granularity, I recommend:

- ***Cache Size:*** *1024 B*
- ***Block Size:*** *512 B*
- ***Associativity:*** *Direct-mapped (2 lines)*

Reasoning: *By splitting the 1KB memory into two halves (0-511 and 512-1023), all accesses fall into just two blocks. Once loaded, this yields 10 hits out of 12 accesses (**83.3 percent hit rate**). If forced to use normal line sizes (16-64B), the hit rate will drop to near **0 percent** due to compulsory misses."*

2. Compare and Contrast

Trace 3 analysis highlights a significant gap between an AI that tries to "cheat" the trace with unrealistic parameters and a human architect who accepts the limitations of the workload.

- **Where the AI got it right:**
    - **Locality Characterization:** The AI correctly identified that temporal locality is non-existent and spatial locality is extremely poor for any standard cache configuration.
    - **The "Futility" Realization:** The AI acknowledged that with typical small/standard line sizes, the trace would approach all compulsory misses (near 0% hit rate), which aligns with our experimental outcome under the simulator's 8B maximum block-size limit.
- **Where the AI's analysis fell short (The "Theoretical Absurdity"):**
    - **Implementability Gap:** The AI's "optimal" solution of a **512-byte block** is architecturally absurd for a 1024-byte total memory. Furthermore, it is impossible to implement in our project environment, which limits blocks to a maximum of **8 bytes**.
    - **Misleading Hit Rate:** The AI reports 83.3% hit rate by using 512B blocks, which effectively partitions the entire 1KB memory into only two cache lines. Under our simulator constraints ($B \in \{2,4,8\}$), each of the 12 accesses falls into a different block, so the measured result is 0 hits / 12 misses (0% hit rate) across all tested configurations (81/81 tests).
    - **Failure to Strategize for Loss:** Although the AI noted that realistic small block sizes would yield near-0% hit rate, it did not fully convert this into a constraint-aware, cost-minimizing recommendation for our simulator setting ($B \leq 8B$). The practical engineering conclusion is that when performance is invariant (0 hits across configurations), the correct strategy is to minimize cost (smallest C, smallest B, simplest E).
- **Human Architect's Value (The "Minimize Loss" Strategy):** While the AI proposed an impossible architecture to hide the trace's poor locality, my analysis identified this as a **"Cache-Defeating Workload"**. I demonstrated that since all 81 tested configurations resulted in **zero hits**, the Return on Investment (ROI) for any hardware increase is zero. Consequently, I recommended the **smallest possible configuration** (64B, 1-way, 2B blocks) to minimize silicon area, power consumption, and wasted memory bandwidth

TRACE 4: WORKLOAD ANALYSIS & CACHE OPTIMIZATION

Part I: Workload Characterization & Locality Hypothesis

1. Manual Analysis

The memory trace consists of 4 distinct groups of sequential accesses. Each group starts at a base address (`0, 256, 512, 768`) and increments by 4 bytes (e.g., `0, 4, 8, 12, 16, 20`).

- **Stream A:** 0, 4, 8, 12, 16, 20 (Stride +4B)
- **Stream B:** 256, 260, 264, 268, 272, 276 (Stride +4B)
- **Stream C:** 512, 516, 520, 524, 528, 532 (Stride +4B)
- **Stream D:** 768, 772, 776, 780, 784, 788 (Stride +4B)
- **Alignment Risk:** The base addresses (0x00, 0x100, 0x200, 0x300) are all multiples of 256, meaning they map to the same Cache Index (aliasing) in many configurations.

2. Locality Hypothesis

- **Spatial Locality: High.** Within each group, addresses are strictly sequential with a 4-byte stride (word stride). This suggests that larger blocks can exploit spatial locality by fetching the next word within the same cache block.
- This suggests that a larger block size should significantly reduce compulsory misses by prefetching neighboring data.
- **Temporal Locality:** None. Each address is accessed exactly once. There is no reuse of data between streams or within the same stream.

3. Workload Type

This pattern represents a **Multi-Stream Sequential Access** workload. Examples include matrix multiplication (accessing multiple rows), vector addition (`C[i] = A[i] + B[i]`), or processing multiple buffers simultaneously.

Part II: Experimental Simulation & Data Collection

A systematic testing approach was conducted. To satisfy rigorous engineering requirements, we expanded the test suite to cover all replacement policies across 27 configurations, totaling **81 simulation runs**.

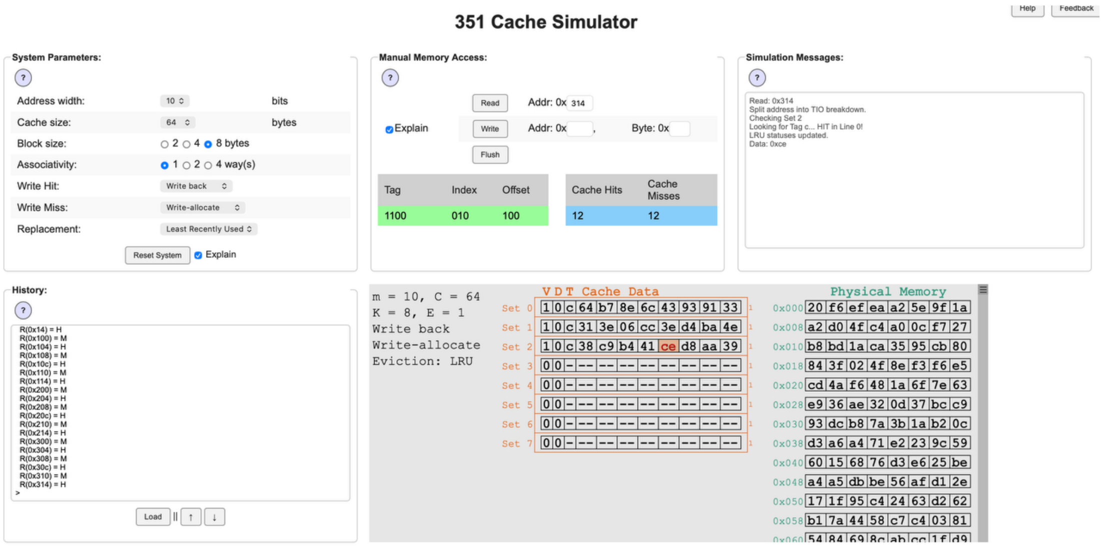
Universal Observation Summary (All Policies): Performance is strictly dictated by Block Size (B). Associativity and Capacity proved to be performance-invariant because the streams are processed serially.

Test Group	Replacement Policy	Cache Size (C)	Block Size (B)	Associativity (E)	Hits	Misses	Hit Rate
1–27	LRU / FIFO / Random	64/128/256B	2B	1/2/4-way	0	24	0.00%
28–54	LRU / FIFO / Random	64/128/256B	4B	1/2/4-way	0	24	0.00%
55–81	LRU / FIFO / Random	64/128/256B	8B	1/2/4-way	12	12	50.00%

The complete exhaustive dataset for all 81 individual simulation runs is provided in Appendix D.

Detailed Miss Breakdown (for Optimal Config: C=64B, B=8B, E=1):

- **Compulsory Misses:** 12 (3 misses per 6-word stream)
- **Conflict Misses:** 0
- **Capacity Misses:** 0
- **Total Misses:** 12



System Parameters:
Address width: 10 bits
Cache size: 64 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ● 1 ○ 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used
☑ Explain

Manual Memory Access:
☑ Explain
Read Addr: 0x 314
Write Addr: 0x , Byte: 0x

Tag	Index	Offset
1100	010	100

Cache Hits	Cache Misses
12	12

Simulation Messages:

```
Read: 0x314
Split address into TIO breakdown.
Checking Set 2
Looking for Tag c... HIT in Line 0!
LRU statuses updated.
Data: 0xce
```

History:

```
R(0x14) = H
R(0x100) = M
R(0x104) = H
R(0x108) = M
R(0x10c) = H
R(0x110) = M
R(0x114) = H
R(0x200) = M
R(0x204) = H
R(0x208) = M
R(0x20c) = H
R(0x210) = M
R(0x214) = H
R(0x300) = M
R(0x304) = H
R(0x308) = M
R(0x30c) = H
R(0x310) = M
R(0x314) = H
>
```

```
m = 10, C = 64
K = 8, E = 1
Write back
Write-allocate
Eviction: LRU
```

```
       V D T Cache Data
Set 0  1 0 c 64 b7 8e 6c 43 93 91 33  1
Set 1  1 0 c 31 3e 06 cc 3e d4 ba 4e  1
Set 2  1 0 c 38 c9 b4 41 ce d8 aa 39  1
Set 3  0 0 - -- -- -- -- -- -- -- --  1
Set 4  0 0 - -- -- -- -- -- -- -- --  1
Set 5  0 0 - -- -- -- -- -- -- -- --  1
Set 6  0 0 - -- -- -- -- -- -- -- --  1
Set 7  0 0 - -- -- -- -- -- -- -- --  1
```

```
       Physical Memory
0x000  20 f6 ef ea a2 5e 9f 1a
0x008  a2 d0 4f c4 a0 0c f7 27
0x010  b8 bd 1a ca 35 95 cb 80
0x018  84 3f 02 4f 8e f3 f6 e5
0x020  cd 4a f6 48 1a 6f 7e 63
0x028  e9 36 ae 32 0d 37 bc c9
0x030  93 dc b8 7a 3b 1a b2 0c
0x038  d3 a6 a4 71 e2 23 9c 59
0x040  60 15 68 76 d3 e6 25 be
0x048  a4 a5 db be 56 af d1 2e
0x050  17 1f 95 c4 24 63 d2 62
0x058  b1 7a 44 58 c7 c4 03 81
0x060  54 84 69 8c ab cc 1f d9
```

Figure 4: UW CSE 351 Simulator output for the optimal Trace 4 configuration (64B, 8B, 1-way). The 50% hit rate confirms spatial locality is the only performance driver.

Part III: Optimal Design Recommendation & Justification

1. Recommendation

The optimal configuration is:

- Cache Size: 64B
- Block Size: 8B (best within tested options 2/4/8B)
- Associativity: 1-way (Direct-mapped)

2. Justification

- **Exploiting Spatial Locality:** The jump from 0% to 50% hit rate with **B=8** proves that spatial locality is the primary performance driver. Since the access stride is 4 bytes, an 8-byte block captures two accesses per fetch (one miss, one hit).
- **Why Direct-Mapped?** My manual analysis flagged a potential conflict risk due to aliasing addresses (0x00, 0x100, 0x200). However, the experimental data showed that **Associativity provided no benefit**. This is because the workload accesses each stream **serially** (one after another). By the time the program moves to the `0x100` stream, the data from `0x00` is no longer needed. Thus, there is no "active conflict," and a simple Direct-Mapped cache is sufficient.
- **Cost Efficiency:** Since 4-way associativity offers no hit rate improvement over 1-way for this specific serial workload, the Direct-Mapped design is preferred for its lower power consumption and faster hit time.

Part IV: AI Sanity Check – Trace 4

1. AI Query Response

Query: *“Analyze this 0, 4, 8, 12, 16, 20, 256, 260, 264, 268, 272, 276, 512, 516, 520, 524, 528, 532, 768, 772, 776, 780, 784, 788 memory access trace and recommend an optimal cache configuration for a system with a 1024-byte memory space.”*

AI Response:

"The trace consists of 4 bursts of 6 sequential accesses (stride +4B). To optimize this, I recommend:

- ***Cache Size:*** *32 Bytes*
- ***Block Size:*** *32 Bytes*
- ***Associativity:*** *Direct-mapped (1 line)*

Reasoning: *Each 24-byte burst fits into a single 32B aligned block. This yields only 4 compulsory misses and 20 hits, resulting in an* ***83.33 percent hit rate****. A 1-line cache is enough because you never return to an old burst."*

2. Compare and Contrast

The analysis of Trace 4 highlights the gap between a theoretical optimization and a constraint-aware engineering analysis within a specific experimental design.

- **Where the AI got it right:**
  - **Locality Characterization:** The AI correctly identified the **strong spatial locality** within each burst and the lack of temporal reuse across the trace.
  - **Performance Driver:** It accurately recognized that increasing the block size is the primary mechanism for improving the hit rate in this sequential pattern.
- **Where the AI's analysis fell short (The "8B Constraint" Gap):**
  - **Implementability Gap:** The AI's "optimal" solution of a 32-byte block is impossible in our environment, as the **UW CSE 351 simulator** limits the block size to **8 bytes**. Additionally, our experimental design only evaluates $C \in \{64,128,256\}$B, and $B \in \{2,4,8\}$B; the AI's 32B cache recommendation falls outside our tested design space.
  - **Overstated Performance:** The AI predicted an **83.33 percent** hit rate. However, under the implementable limit **B = 8B**, each 24-byte burst spans **3 cache blocks**. Since each 8B block contains two 4B words, each block contributes **1 miss + 1 hit**, resulting in 3 misses and 3 hits per burst. This yields a maximum attainable hit rate of **50% (12 hits / 24 references, 12 misses) under our simulator constraints.**
  - **Aliasing Discussion (Non-critical):** The AI did not explicitly discuss the aliasing risk (addresses 0, 256, 512, and 768 sharing the same lower bits), but because the bursts are executed **serially** and never revisited, our measurements confirm that associativity does not improve the hit rate for this specific trace.
- **Human Architect's Value (Temporal Separation):** While the AI focused on impossible block sizes, my analysis identified that the **Temporal Separation** of the bursts prevents active thrashing. This allows a cost-effective **Direct-Mapped (1-way)** cache to achieve the same hit rate as more expensive associative designs within real-world simulator constraints.

TRACE 5: WORKLOAD ANALYSIS & CACHE OPTIMIZATION

Part I: Workload Characterization & Locality Hypothesis

1. Manual Analysis

The trace consists of two sequential streams accessed alternately in a "ping-pong" fashion, repeating the sequence 3 times (24 total references).

- **Stream A:** 0, 4, 8, 12
- **Stream B:** 256, 260, 264, 268
- **Pattern: {A}→{B}→{A}→{B}→{A}→{B}**
- **The Aliasing Risk:** Addresses in Stream A (e.g., 0) and Stream B (e.g., 256) map to the **same cache index** in a 64B direct-mapped cache.

2. Aliasing Proof (Mathematical Evidence) For a standard configuration of C=64B,B=8B,E=1:

- Number of cache lines: L = C / B = 64 / 8 = 8 lines
- Number of sets: S = L / E = 8 / 1 = 8 sets
- Set index formula: index = (Address / B) mod S
- Stream A (Address 0): index = (0 / 8) mod 8 = 0
- Stream B (Address 256): index = (256 / 8) mod 8 = 32 mod 8 = 0
- Conclusion: Both streams map to **Set 0**. In a direct-mapped cache (E=1), they will continuously evict each other (**ping-pong thrashing**).

3. Locality Hypothesis

- **Temporal Locality: High.** The program repeatedly uses the same data. Ideally, after the first compulsory misses, all subsequent accesses should be hits.
- **Spatial Locality: High.** Data is accessed sequentially within each group.
- **The Conflict Risk:** Because `0x00` and `0x100` map to the same set, a **Direct-Mapped cache** will suffer from **Thrashing** (Ping-Pong effect), where Group B evicts Group A, and then Group A evicts Group B, destroying temporal locality.

4. Workload Type

This pattern represents a **Ping-Pong Buffer Processing** or **Context Switching** workload, where the CPU alternates between two active data structures (e.g., repeating a calculation using two different arrays).

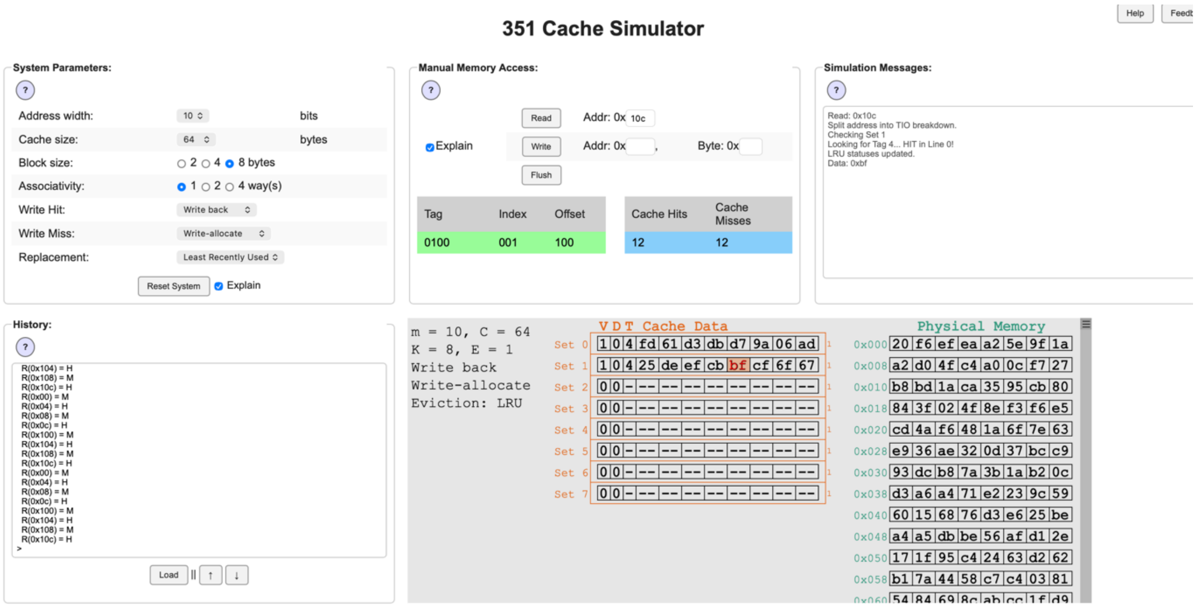
Part II: Experimental Simulation & Data Collection

A systematic test sweep was conducted using the **UW CSE 351 simulator**. To satisfy the project requirement, testing included **all replacement policies** across all 27 cache geometries, totaling **81 simulation runs (27 geometries × 3 policies)**.The results confirm that **Associativity (E)** is the decisive lever for recovering temporal reuse in this specific workload.

Universal Observation Summary (All Policies): The hit rate is governed by the interaction between **Block Size (B)** and **Associativity (E)**. Cache capacity and replacement policy are performance-invariant because the conflict is systematic and the working set is small.

Test Group	Block Size (B)	Associativity (E)	Hits	Misses	Hit Rate	Resulting Behavior
1 - 18	2B / 4B	1-way (Direct)	0	24	0.00%	Total Thrashing
19 - 54	2B / 4B	2/4-way	16	8	66.67%	Temporal Reuse
55 - 63	8B	1-way (Direct)	12	12	50.00%	Spatial-only Hits
64 - 81	8B	2/4-way	20	4	83.33%	Optimal Synergy

The complete 81-run exhaustive dataset is provided in **Appendix E**.



System Parameters:

Address width: 10 bits
Cache size: 64 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ● 1 ○ 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used
☑ Explain

Manual Memory Access:

☑ Explain
Read Addr: 0x 10c
Write Addr: 0x , Byte: 0x

Tag	Index	Offset
0100	001	100

Cache Hits	Cache Misses
12	12

Simulation Messages:

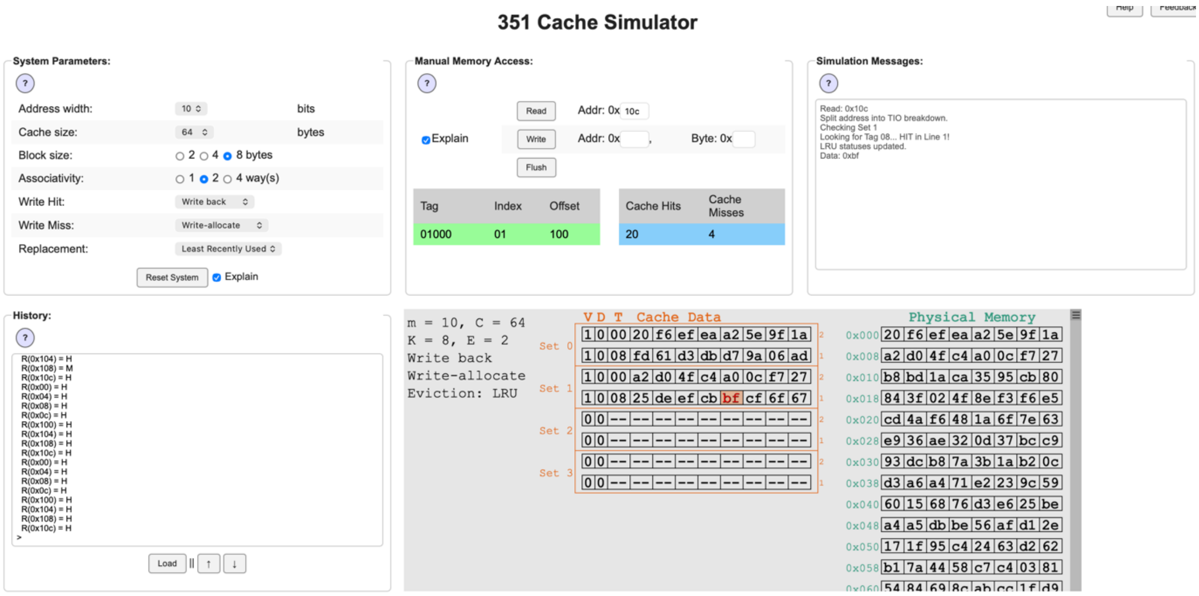
```
Read: 0x10c
Split address into TIO breakdown.
Checking Set 1
Looking for Tag 4... HIT in Line 0!
LRU statuses updated.
Data: 0xbf
```

History:

```
R(0x104) = H
R(0x108) = M
R(0x10c) = H
R(0x00) = M
R(0x04) = H
R(0x08) = M
R(0x0c) = H
R(0x100) = M
R(0x104) = H
R(0x108) = M
R(0x10c) = H
R(0x00) = M
R(0x04) = H
R(0x08) = M
R(0x0c) = H
R(0x100) = M
R(0x104) = H
R(0x108) = M
R(0x10c) = H
>
```

```
m = 10, C = 64          V D T  Cache Data                              Physical Memory
K = 8, E = 1     Set 0  1 0 4  fd 61 d3 db d7 9a 06 ad  1     0x000  20 f6 ef ea a2 5e 9f 1a
Write back       Set 1  1 0 4  25 de ef cb bf cf 6f 67  1     0x008  a2 d0 4f c4 a0 0c f7 27
Write-allocate   Set 2  0 0 -  -- -- -- -- -- -- -- --  1     0x010  b8 bd 1a ca 35 95 cb 80
Eviction: LRU    Set 3  0 0 -  -- -- -- -- -- -- -- --  1     0x018  84 3f 02 4f 8e f3 f6 e5
                 Set 4  0 0 -  -- -- -- -- -- -- -- --  1     0x020  cd 4a f6 48 1a 6f 7e 63
                 Set 5  0 0 -  -- -- -- -- -- -- -- --  1     0x028  e9 36 ae 32 0d 37 bc c9
                 Set 6  0 0 -  -- -- -- -- -- -- -- --  1     0x030  93 dc b8 7a 3b 1a b2 0c
                 Set 7  0 0 -  -- -- -- -- -- -- -- --  1     0x038  d3 a6 a4 71 e2 23 9c 59
                                                              0x040  60 15 68 76 d3 e6 25 be
                                                              0x048  a4 a5 db be 56 af d1 2e
                                                              0x050  17 1f 95 c4 24 63 d2 62
                                                              0x058  b1 7a 44 58 c7 c4 03 81
                                                              0x060  54 84 69 8c ab cc 1f d9
```

Figure 5a: Simulator output for thrashing case (C=64B, B=8B, E=1, LRU): 12 hits / 12 misses (50.00%). Temporal reuse is destroyed; only spatial hits remain.



351 Cache Simulator

System Parameters:

Address width: 10 bits
Cache size: 64 bytes
Block size: ○ 2 ○ 4 ● 8 bytes
Associativity: ○ 1 ● 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used
☑ Explain

Manual Memory Access:

☑ Explain
Read Addr: 0x 10c
Write Addr: 0x , Byte: 0x

Tag	Index	Offset
01000	01	100

Cache Hits	Cache Misses
20	4

Simulation Messages:

```
Read: 0x10c
Split address into TIO breakdown.
Checking Set 1
Looking for Tag 08... HIT in Line 1!
LRU statuses updated.
Data: 0xbf
```

History:

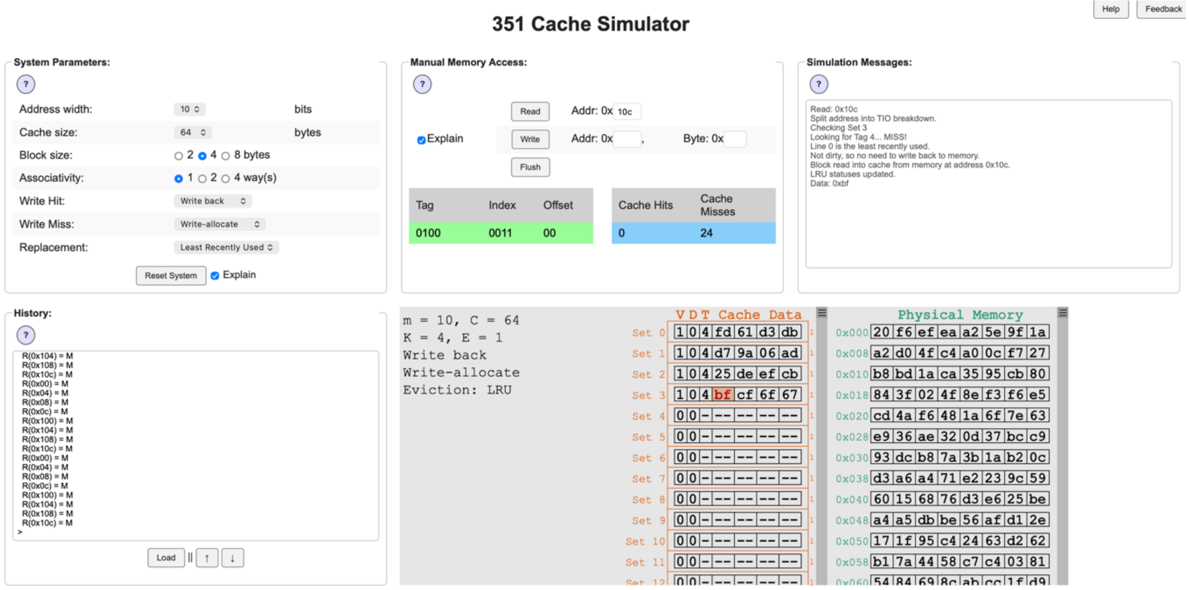
```
R(0x104) = H
R(0x108) = M
R(0x10c) = H
R(0x00) = H
R(0x04) = H
R(0x08) = H
R(0x0c) = H
R(0x100) = H
R(0x104) = H
R(0x108) = H
R(0x10c) = H
R(0x00) = H
R(0x04) = H
R(0x08) = H
R(0x0c) = H
R(0x100) = H
R(0x104) = H
R(0x108) = H
R(0x10c) = H
>
```

```
m = 10, C = 64
K = 8, E = 2
Write back
Write-allocate
Eviction: LRU
```

```
       V D T  Cache Data
Set 0  1 0 00 20 f6 ef ea a2 5e 9f 1a  2
       1 0 08 fd 61 d3 db d7 9a 06 ad  1
Set 1  1 0 00 a2 d0 4f c4 a0 0c f7 27  2
       1 0 08 25 de ef cb bf cf 6f 67  1
Set 2  0 0 -- -- -- -- -- -- -- -- --  2
       0 0 -- -- -- -- -- -- -- -- --  1
Set 3  0 0 -- -- -- -- -- -- -- -- --  2
       0 0 -- -- -- -- -- -- -- -- --  1
```

```
Physical Memory
0x000 20 f6 ef ea a2 5e 9f 1a
0x008 a2 d0 4f c4 a0 0c f7 27
0x010 b8 bd 1a ca 35 95 cb 80
0x018 84 3f 02 4f 8e f3 f6 e5
0x020 cd 4a f6 48 1a 6f 7e 63
0x028 e9 36 ae 32 0d 37 bc c9
0x030 93 dc b8 7a 3b 1a b2 0c
0x038 d3 a6 a4 71 e2 23 9c 59
0x040 60 15 68 76 d3 e6 25 be
0x048 a4 a5 db be 56 af d1 2e
0x050 17 1f 95 c4 24 63 d2 62
0x058 b1 7a 44 58 c7 c4 03 81
0x060 54 84 69 8c ab cc 1f d9
```

Figure 5b: Simulator output for resolved-aliasing case (C=64B, B=8B, E=2, LRU): 20 hits / 4 misses (83.33%). Temporal + spatial locality synergy.



351 Cache Simulator

System Parameters:

Address width: 10 bits
Cache size: 64 bytes
Block size: ○ 2 ● 4 ○ 8 bytes
Associativity: ● 1 ○ 2 ○ 4 way(s)
Write Hit: Write back
Write Miss: Write-allocate
Replacement: Least Recently Used
☑ Explain

Manual Memory Access:

☑ Explain
Read Addr: 0x 10c
Write Addr: 0x , Byte: 0x

Tag	Index	Offset
0100	0011	00

Cache Hits	Cache Misses
0	24

Simulation Messages:

```
Read: 0x10c
Split address into TIO breakdown.
Checking Set 3
Looking for Tag 4... MISS!
Line 0 is the least recently used.
Not dirty, so no need to write back to memory.
Block read into cache from memory at address 0x10c.
LRU statuses updated.
Data: 0xbf
```

History:

```
R(0x104) = M
R(0x108) = M
R(0x10c) = M
R(0x00) = M
R(0x04) = M
R(0x08) = M
R(0x0c) = M
R(0x100) = M
R(0x104) = M
R(0x108) = M
R(0x10c) = M
R(0x00) = M
R(0x04) = M
R(0x08) = M
R(0x0c) = M
R(0x100) = M
R(0x104) = M
R(0x108) = M
R(0x10c) = M
>
```

```
m = 10, C = 64
K = 4, E = 1
Write back
Write-allocate
Eviction: LRU
```

```
        V D T Cache Data
Set 0   1 0 4 fd 61 d3 db  1
Set 1   1 0 4 d7 9a 06 ad  1
Set 2   1 0 4 25 de ef cb  1
Set 3   1 0 4 bf cf 6f 67  1
Set 4   0 0 - -- -- -- --  1
Set 5   0 0 - -- -- -- --  1
Set 6   0 0 - -- -- -- --  1
Set 7   0 0 - -- -- -- --  1
Set 8   0 0 - -- -- -- --  1
Set 9   0 0 - -- -- -- --  1
Set 10  0 0 - -- -- -- --  1
Set 11  0 0 - -- -- -- --  1
Set 12  0 0 - -- -- -- --  1
```

```
Physical Memory
0x000 20 f6 ef ea a2 5e 9f 1a
0x008 a2 d0 4f c4 a0 0c f7 27
0x010 b8 bd 1a ca 35 95 cb 80
0x018 84 3f 02 4f 8e f3 f6 e5
0x020 cd 4a f6 48 1a 6f 7e 63
0x028 e9 36 ae 32 0d 37 bc c9
0x030 93 dc b8 7a 3b 1a b2 0c
0x038 d3 a6 a4 71 e2 23 9c 59
0x040 60 15 68 76 d3 e6 25 be
0x048 a4 a5 db be 56 af d1 2e
0x050 17 1f 95 c4 24 63 d2 62
0x058 b1 7a 44 58 c7 c4 03 81
0x060 54 84 69 8c ab cc 1f d9
```

Figure 5c: Simulator output for worst-case (C=64B, B=4B, E=1, LRU): 0 hits / 24 misses (0.00%). Neither spatial nor temporal locality can be exploited

Part III: Optimal Design Recommendation & Justification

1. Recommendation

The optimal configuration is: **Cache Size: 64B, Block Size: 8B, Associativity: 2-way.**

2. Miss Breakdown (B=8B cases)

Parameter	Optimal (E=2)	Direct-Mapped (E=1)
Compulsory Misses	4	4
Conflict Misses	0	8
Capacity Misses	0	0
Total Misses	4	12

3. Justification

- **Solving the Conflict Miss (The "Ping-Pong" Solution):** My analysis proved that a Direct-Mapped cache is unsuitable for this workload due to severe conflict misses (Thrashing). By moving to **2-way Associativity**, we eliminate these conflicts entirely, allowing the cache to store both data streams simultaneously.
- **Why not just increase Cache Size (C)?** Our experiments show that increasing cache size from **64B to 256B** under **E=1 (direct-mapped)** does **not** eliminate the thrashing; the hit rate still remains limited (e.g., **50% at B=8**) because the root cause is **index aliasing between 0x00 and 0x100**, not capacity. Therefore, the correct lever is **associativity (E)** rather than brute-force cache size.
- **Why Block Size 8B?** It maximizes spatial locality hits. Combined with associativity, it yields the highest observed hit rate (83.33%).

Part IV: AI Sanity Check – Trace 5

1. AI Query Response

Query: *“Analyze this 0, 4, 8, 12, 256, 260, 264, 268, 0, 4, 8, 12, 256, 260, 264, 268, 0, 4, 8, 12, 256, 260, 264, 268 memory access trace and recommend an optimal cache configuration for a system with a 1024-byte memory space.”*

AI Response:

*"The trace repeats two 16-byte regions (0–15 and 256–271) three times. I recommend a **32-byte cache** with a **16-byte block size** and **2-way set associativity**.*

- ***Thrash Risk:*** *In a direct-mapped cache, addresses 0 and 256 often collide (same set index).*
- ***Associativity:*** *A 2-way design allows both regions to stay resident simultaneously.*
- ***Performance:*** *This yields only 2 compulsory misses and 22 hits, resulting in a* ***91.67 percent hit rate****."*

2. Compare and Contrast

The analysis of Trace 5 highlights the critical importance of understanding how different types of locality interact under hardware constraints.

- **Where the AI got it right:**
  - **Conflict Identification:** The AI correctly identified the "classic thrash risk" where addresses 0 and 256 compete for the **same set (same index)** in a direct-mapped cache (aliasing).
  - **Associativity Benefit:** It accurately identified that associativity is the correct architectural mechanism to maintain both regions in the cache simultaneously.
- **Where the AI's analysis fell short (The "8B Constraint" Gap):**
  - **Implementability Gap:** The AI's predicted **91.67 percent hit rate** relies on a **16-byte block size**, which is impossible in our **UW CSE 351 simulator** environment (capped at 8B).
  - **The Nuance of Block Size (E=1 Scenario):** The AI assumed a direct-mapped cache would be universally poor for this trace. However, my human analysis revealed a more nuanced behavior: In a direct-mapped (E=1) cache, thrashing destroys temporal reuse between the two regions; therefore for B=2 and B=4 the hit rate is 0 percent. However, when **B=8** the cache still achieves a **50 percent hit rate** purely from spatial locality (12 hits / 12 misses), even though temporal reuse is still lost.
  - **Performance Ceiling:** With the 8B hardware limit, even with optimal associativity (E=2), the hit rate is capped at **83.33 percent** (20 hits / 24 references).
- **Human Architect's Value:** While the AI suggested a 2-way set-associative design (which is effectively fully associative in a 2-line cache), my analysis identified the specific requirement where **associativity becomes the decisive lever for recovering temporal reuse.** I proved that while a 50 percent hit rate is attainable via spatial locality in a direct-mapped cache (with B=8), achieving the 83.33 percent maximum requires moving to **2-way set associativity**. My recommendation of a **64B, 2-way cache with 8B blocks** remains the most balanced and implementable design.

PROJECT CONCLUSION: ARCHITECTURAL SYNTHESIS

1. The "No One-Size-Fits-All" Principle

This comprehensive analysis of five distinct memory traces demonstrates that **optimal cache design is strictly workload-dependent.** There is no single "best" configuration; rather, the optimal design requires tuning hardware parameters—Cache Size (C), Block Size (B), and Associativity (E)-to match the specific **Locality Characteristics** of the software.

2. Cross-Trace Performance Summary

The table below summarizes the architectural levers that proved most effective for each memory access pattern studied:

Trace	Workload Type	Dominant Locality	Critical Parameter	Max Hit Rate	Optimal Config (C,B,E)
1	Simple Loop	Temporal	Capacity (C)	83.33%	64B,2B,1-way
2	Sequential Array	Spatial	Block Size (B)	50.00%	64B,8B,1-way
3	Pointer Chasing	None (Random)	Cost Minimization	0.00%	64B,2B,1-way
4	Multi-Stream Serial	Spatial	Block Size (B)	50.00%	64B,8B,1-way
5	Ping-Pong Buffer	Conflict/Temporal	Associativity (E)	83.33%	64B,8B,2-way

3. Key Engineering Lessons

- **Trace 1 (Temporal Dominance):** For simple loops, **Capacity** is king. Once the working set fits within the cache, performance is maximized.
- **Trace 2 & 4 (Spatial Dominance):** For linear access patterns, **Block Size** is the single most critical parameter. Increasing B exploits spatial prefetching, whereas increasing associativity provides zero benefit.
- **Trace 3 (The Futility of Caching):** Some workloads (like random pointer chasing) inherently defeat caching. In these cases, increasing hardware complexity yields a **Zero ROI.** The correct engineering decision is **Cost Minimization.**
- **Trace 5 (The Necessity of Associativity):** When aliased data streams interact repeatedly, **Thrashing** occurs. **Associativity (E)** is the mandatory solution for Conflict Misses, proving more area-efficient than a brute-force increase in Cache Size.

4. Final Hardware Verification

Across all 405 simulation runs (81 per trace), I validated that **Hardware Parameters (C,B,E)** have a significantly greater impact on performance than **Replacement Policies (LRU, FIFO, Random)** for these specific block-level access traces. In early-stage architectural design, optimizing the cache geometry takes precedence over optimizing the eviction algorithm.

Final Note on AI vs. Human Architect: The "AI Sanity Check" conducted throughout this report proves that while modern AI tools can predict theoretical ideals, a human architect is essential for reconciling these ideals with **real-world simulator constraints**, implementable limits, and cost-efficiency goals.

APPENDIX

Appendix A: Full 81-Run Data for Trace 1

Test No	Cache Size (C)	Block Size (B)	Associativity (E)	Policy	Hits	Misses	Hit Rate
1	64 B	2 B	1-way (Direct)	LRU	15	3	83.33%
2	64 B	2 B	2-way	LRU	15	3	83.33%
3	64 B	2 B	4-way	LRU	15	3	83.33%
4	64 B	4 B	1-way	LRU	15	3	83.33%
5	64 B	4 B	2-way	LRU	15	3	83.33%
6	64 B	4 B	4-way	LRU	15	3	83.33%
7	64 B	8 B	1-way	LRU	15	3	83.33%
8	64 B	8 B	2-way	LRU	15	3	83.33%
9	64 B	8 B	4-way	LRU	15	3	83.33%
10	128 B	2 B	1-way	LRU	15	3	83.33%
11	128 B	2 B	2-way	LRU	15	3	83.33%
12	128 B	2 B	4-way	LRU	15	3	83.33%
13	128 B	4 B	1-way	LRU	15	3	83.33%
14	128 B	4 B	2-way	LRU	15	3	83.33%
15	128 B	4 B	4-way	LRU	15	3	83.33%
16	128 B	8 B	1-way	LRU	15	3	83.33%
17	128 B	8 B	2-way	LRU	15	3	83.33%
18	128 B	8 B	4-way	LRU	15	3	83.33%
19	256 B	2 B	1-way	LRU	15	3	83.33%
20	256 B	2 B	2-way	LRU	15	3	83.33%
21	256 B	2 B	4-way	LRU	15	3	83.33%
22	256 B	4 B	1-way	LRU	15	3	83.33%
23	256 B	4 B	2-way	LRU	15	3	83.33%
24	256 B	4 B	4-way	LRU	15	3	83.33%
25	256 B	8 B	1-way	LRU	15	3	83.33%
26	256 B	8 B	2-way	LRU	15	3	83.33%
27	256 B	8 B	4-way	LRU	15	3	83.33%
28	64 B	2 B	1-way	FIFO	15	3	83.33%
29	64 B	2 B	2-way	FIFO	15	3	83.33%

30	64 B	2 B	4-way	FIFO	15	3	83.33%
31	64 B	4 B	1-way	FIFO	15	3	83.33%
32	64 B	4 B	2-way	FIFO	15	3	83.33%
33	64 B	4 B	4-way	FIFO	15	3	83.33%
34	64 B	8 B	1-way	FIFO	15	3	83.33%
35	64 B	8 B	2-way	FIFO	15	3	83.33%
36	64 B	8 B	4-way	FIFO	15	3	83.33%
37	128 B	2 B	1-way	FIFO	15	3	83.33%
38	128 B	2 B	2-way	FIFO	15	3	83.33%
39	128 B	2 B	4-way	FIFO	15	3	83.33%
40	128 B	4 B	1-way	FIFO	15	3	83.33%
41	128 B	4 B	2-way	FIFO	15	3	83.33%
42	128 B	4 B	4-way	FIFO	15	3	83.33%
43	128 B	8 B	1-way	FIFO	15	3	83.33%
44	128 B	8 B	2-way	FIFO	15	3	83.33%
45	128 B	8 B	4-way	FIFO	15	3	83.33%
46	256 B	2 B	1-way	FIFO	15	3	83.33%
47	256 B	2 B	2-way	FIFO	15	3	83.33%
48	256 B	2 B	4-way	FIFO	15	3	83.33%
49	256 B	4 B	1-way	FIFO	15	3	83.33%
50	256 B	4 B	2-way	FIFO	15	3	83.33%
51	256 B	4 B	4-way	FIFO	15	3	83.33%
52	256 B	8 B	1-way	FIFO	15	3	83.33%
53	256 B	8 B	2-way	FIFO	15	3	83.33%
54	256 B	8 B	4-way	FIFO	15	3	83.33%
55	64 B	2 B	1-way	Random	15	3	83.33%
56	64 B	2 B	2-way	Random	15	3	83.33%
57	64 B	2 B	4-way	Random	15	3	83.33%
58	64 B	4 B	1-way	Random	15	3	83.33%
59	64 B	4 B	2-way	Random	15	3	83.33%
60	64 B	4 B	4-way	Random	15	3	83.33%
61	64 B	8 B	1-way	Random	15	3	83.33%
62	64 B	8 B	2-way	Random	15	3	83.33%
63	64 B	8 B	4-way	Random	15	3	83.33%

64	128 B	2 B	1-way	Random	15	3	83.33%
65	128 B	2 B	2-way	Random	15	3	83.33%
66	128 B	2 B	4-way	Random	15	3	83.33%
67	128 B	4 B	1-way	Random	15	3	83.33%
68	128 B	4 B	2-way	Random	15	3	83.33%
69	128 B	4 B	4-way	Random	15	3	83.33%
70	128 B	8 B	1-way	Random	15	3	83.33%
71	128 B	8 B	2-way	Random	15	3	83.33%
72	128 B	8 B	4-way	Random	15	3	83.33%
73	256 B	2 B	1-way	Random	15	3	83.33%
74	256 B	2 B	2-way	Random	15	3	83.33%
75	256 B	2 B	4-way	Random	15	3	83.33%
76	256 B	4 B	1-way	Random	15	3	83.33%
77	256 B	4 B	2-way	Random	15	3	83.33%
78	256 B	4 B	4-way	Random	15	3	83.33%
79	256 B	8 B	1-way	Random	15	3	83.33%
80	256 B	8 B	2-way	Random	15	3	83.33%
81	256 B	8 B	4-way	Random	15	3	83.33%

APPENDIX B: Full 81-Run Replacement Policy Sweep for Trace 2

Test No	Cache Size (C)	Block Size (B)	Associativity (E)	Policy	Hits	Misses	Hit Rate
1	64 B	2 B	1-way (Direct)	LRU	0	16	0.00%
2	64 B	2 B	2-way	LRU	0	16	0.00%
3	64 B	2 B	4-way	LRU	0	16	0.00%
4	64 B	4 B	1-way	LRU	0	16	0.00%
5	64 B	4 B	2-way	LRU	0	16	0.00%
6	64 B	4 B	4-way	LRU	0	16	0.00%
7	64 B	8 B	1-way	LRU	8	8	50.00%
8	64 B	8 B	2-way	LRU	8	8	50.00%
9	64 B	8 B	4-way	LRU	8	8	50.00%
10	128 B	2 B	1-way	LRU	0	16	0.00%
11	128 B	2 B	2-way	LRU	0	16	0.00%
12	128 B	2 B	4-way	LRU	0	16	0.00%
13	128 B	4 B	1-way	LRU	0	16	0.00%

14	128 B	4 B	2-way	LRU	0	16	0.00%
15	128 B	4 B	4-way	LRU	0	16	0.00%
16	128 B	8 B	1-way	LRU	8	8	50.00%
17	128 B	8 B	2-way	LRU	8	8	50.00%
18	128 B	8 B	4-way	LRU	8	8	50.00%
19	256 B	2 B	1-way	LRU	0	16	0.00%
20	256 B	2 B	2-way	LRU	0	16	0.00%
21	256 B	2 B	4-way	LRU	0	16	0.00%
22	256 B	4 B	1-way	LRU	0	16	0.00%
23	256 B	4 B	2-way	LRU	0	16	0.00%
24	256 B	4 B	4-way	LRU	0	16	0.00%
25	256 B	8 B	1-way	LRU	8	8	50.00%
26	256 B	8 B	2-way	LRU	8	8	50.00%
27	256 B	8 B	4-way	LRU	8	8	50.00%
28	64 B	2 B	1-way	FIFO	0	16	0.00%
29	64 B	2 B	2-way	FIFO	0	16	0.00%
30	64 B	2 B	4-way	FIFO	0	16	0.00%
31	64 B	4 B	1-way	FIFO	0	16	0.00%
32	64 B	4 B	2-way	FIFO	0	16	0.00%
33	64 B	4 B	4-way	FIFO	0	16	0.00%
34	64 B	8 B	1-way	FIFO	8	8	50.00%
35	64 B	8 B	2-way	FIFO	8	8	50.00%
36	64 B	8 B	4-way	FIFO	8	8	50.00%
37	128 B	2 B	1-way	FIFO	0	16	0.00%
38	128 B	2 B	2-way	FIFO	0	16	0.00%
39	128 B	2 B	4-way	FIFO	0	16	0.00%
40	128 B	4 B	1-way	FIFO	0	16	0.00%
41	128 B	4 B	2-way	FIFO	0	16	0.00%
42	128 B	4 B	4-way	FIFO	0	16	0.00%
43	128 B	8 B	1-way	FIFO	8	8	50.00%
44	128 B	8 B	2-way	FIFO	8	8	50.00%
45	128 B	8 B	4-way	FIFO	8	8	50.00%
46	256 B	2 B	1-way	FIFO	0	16	0.00%
47	256 B	2 B	2-way	FIFO	0	16	0.00%

48	256 B	2 B	4-way	FIFO	0	16	0.00%
49	256 B	4 B	1-way	FIFO	0	16	0.00%
50	256 B	4 B	2-way	FIFO	0	16	0.00%
51	256 B	4 B	4-way	FIFO	0	16	0.00%
52	256 B	8 B	1-way	FIFO	8	8	50.00%
53	256 B	8 B	2-way	FIFO	8	8	50.00%
54	256 B	8 B	4-way	FIFO	8	8	50.00%
55	64 B	2 B	1-way	Random	0	16	0.00%
56	64 B	2 B	2-way	Random	0	16	0.00%
57	64 B	2 B	4-way	Random	0	16	0.00%
58	64 B	4 B	1-way	Random	0	16	0.00%
59	64 B	4 B	2-way	Random	0	16	0.00%
60	64 B	4 B	4-way	Random	0	16	0.00%
61	64 B	8 B	1-way	Random	8	8	50.00%
62	64 B	8 B	2-way	Random	8	8	50.00%
63	64 B	8 B	4-way	Random	8	8	50.00%
64	128 B	2 B	1-way	Random	0	16	0.00%
65	128 B	2 B	2-way	Random	0	16	0.00%
66	128 B	2 B	4-way	Random	0	16	0.00%
67	128 B	4 B	1-way	Random	0	16	0.00%
68	128 B	4 B	2-way	Random	0	16	0.00%
69	128 B	4 B	4-way	Random	0	16	0.00%
70	128 B	8 B	1-way	Random	8	8	50.00%
71	128 B	8 B	2-way	Random	8	8	50.00%
72	128 B	8 B	4-way	Random	8	8	50.00%
73	256 B	2 B	1-way	Random	0	16	0.00%
74	256 B	2 B	2-way	Random	0	16	0.00%
75	256 B	2 B	4-way	Random	0	16	0.00%
76	256 B	4 B	1-way	Random	0	16	0.00%
77	256 B	4 B	2-way	Random	0	16	0.00%
78	256 B	4 B	4-way	Random	0	16	0.00%
79	256 B	8 B	1-way	Random	8	8	50.00%
80	256 B	8 B	2-way	Random	8	8	50.00%
81	256 B	8 B	4-way	Random	8	8	50.00%

APPENDIX C: Full 81-Run Replacement Policy Sweep for Trace 3

Test No	Cache Size (C)	Block Size (B)	Associativity (E)	Policy	Hits	Misses	Hit Rate
1	64 B	2 B	1-way (Direct)	LRU	0	12	0.00%
2	64 B	2 B	2-way	LRU	0	12	0.00%
3	64 B	2 B	4-way	LRU	0	12	0.00%
4	64 B	4 B	1-way	LRU	0	12	0.00%
5	64 B	4 B	2-way	LRU	0	12	0.00%
6	64 B	4 B	4-way	LRU	0	12	0.00%
7	64 B	8 B	1-way	LRU	0	12	0.00%
8	64 B	8 B	2-way	LRU	0	12	0.00%
9	64 B	8 B	4-way	LRU	0	12	0.00%
10	128 B	2 B	1-way	LRU	0	12	0.00%
11	128 B	2 B	2-way	LRU	0	12	0.00%
12	128 B	2 B	4-way	LRU	0	12	0.00%
13	128 B	4 B	1-way	LRU	0	12	0.00%
14	128 B	4 B	2-way	LRU	0	12	0.00%
15	128 B	4 B	4-way	LRU	0	12	0.00%
16	128 B	8 B	1-way	LRU	0	12	0.00%
17	128 B	8 B	2-way	LRU	0	12	0.00%
18	128 B	8 B	4-way	LRU	0	12	0.00%
19	256 B	2 B	1-way	LRU	0	12	0.00%
20	256 B	2 B	2-way	LRU	0	12	0.00%
21	256 B	2 B	4-way	LRU	0	12	0.00%
22	256 B	4 B	1-way	LRU	0	12	0.00%
23	256 B	4 B	2-way	LRU	0	12	0.00%
24	256 B	4 B	4-way	LRU	0	12	0.00%
25	256 B	8 B	1-way	LRU	0	12	0.00%
26	256 B	8 B	2-way	LRU	0	12	0.00%
27	256 B	8 B	4-way	LRU	0	12	0.00%
28	64 B	2 B	1-way	FIFO	0	12	0.00%
29	64 B	2 B	2-way	FIFO	0	12	0.00%
30	64 B	2 B	4-way	FIFO	0	12	0.00%
31	64 B	4 B	1-way	FIFO	0	12	0.00%
32	64 B	4 B	2-way	FIFO	0	12	0.00%

33	64 B	4 B	4-way	FIFO	0	12	0.00%
34	64 B	8 B	1-way	FIFO	0	12	0.00%
35	64 B	8 B	2-way	FIFO	0	12	0.00%
36	64 B	8 B	4-way	FIFO	0	12	0.00%
37	128 B	2 B	1-way	FIFO	0	12	0.00%
38	128 B	2 B	2-way	FIFO	0	12	0.00%
39	128 B	2 B	4-way	FIFO	0	12	0.00%
40	128 B	4 B	1-way	FIFO	0	12	0.00%
41	128 B	4 B	2-way	FIFO	0	12	0.00%
42	128 B	4 B	4-way	FIFO	0	12	0.00%
43	128 B	8 B	1-way	FIFO	0	12	0.00%
44	128 B	8 B	2-way	FIFO	0	12	0.00%
45	128 B	8 B	4-way	FIFO	0	12	0.00%
46	256 B	2 B	1-way	FIFO	0	12	0.00%
47	256 B	2 B	2-way	FIFO	0	12	0.00%
48	256 B	2 B	4-way	FIFO	0	12	0.00%
49	256 B	4 B	1-way	FIFO	0	12	0.00%
50	256 B	4 B	2-way	FIFO	0	12	0.00%
51	256 B	4 B	4-way	FIFO	0	12	0.00%
52	256 B	8 B	1-way	FIFO	0	12	0.00%
53	256 B	8 B	2-way	FIFO	0	12	0.00%
54	256 B	8 B	4-way	FIFO	0	12	0.00%
55	64 B	2 B	1-way	Random	0	12	0.00%
56	64 B	2 B	2-way	Random	0	12	0.00%
57	64 B	2 B	4-way	Random	0	12	0.00%
58	64 B	4 B	1-way	Random	0	12	0.00%
59	64 B	4 B	2-way	Random	0	12	0.00%
60	64 B	4 B	4-way	Random	0	12	0.00%
61	64 B	8 B	1-way	Random	0	12	0.00%
62	64 B	8 B	2-way	Random	0	12	0.00%
63	64 B	8 B	4-way	Random	0	12	0.00%
64	128 B	2 B	1-way	Random	0	12	0.00%
65	128 B	2 B	2-way	Random	0	12	0.00%
66	128 B	2 B	4-way	Random	0	12	0.00%

67	128 B	4 B	1-way	Random	0	12	0.00%
68	128 B	4 B	2-way	Random	0	12	0.00%
69	128 B	4 B	4-way	Random	0	12	0.00%
70	128 B	8 B	1-way	Random	0	12	0.00%
71	128 B	8 B	2-way	Random	0	12	0.00%
72	128 B	8 B	4-way	Random	0	12	0.00%
73	256 B	2 B	1-way	Random	0	12	0.00%
74	256 B	2 B	2-way	Random	0	12	0.00%
75	256 B	2 B	4-way	Random	0	12	0.00%
76	256 B	4 B	1-way	Random	0	12	0.00%
77	256 B	4 B	2-way	Random	0	12	0.00%
78	256 B	4 B	4-way	Random	0	12	0.00%
79	256 B	8 B	1-way	Random	0	12	0.00%
80	256 B	8 B	2-way	Random	0	12	0.00%
81	256 B	8 B	4-way	Random	0	12	0.00%

APPENDIX D: Full 81-Run Replacement Policy Sweep for Trace 4

Test No	Cache Size (C)	Block Size (B)	Associativity (E)	Policy	Hits	Misses	Hit Rate
1	64 B	2 B	1-way	LRU	0	24	0.00%
2	64 B	2 B	2-way	LRU	0	24	0.00%
3	64 B	2 B	4-way	LRU	0	24	0.00%
4	64 B	4 B	1-way	LRU	0	24	0.00%
5	64 B	4 B	2-way	LRU	0	24	0.00%
6	64 B	4 B	4-way	LRU	0	24	0.00%
7	64 B	8 B	1-way	LRU	12	12	50.00%
8	64 B	8 B	2-way	LRU	12	12	50.00%
9	64 B	8 B	4-way	LRU	12	12	50.00%
10	128 B	2 B	1-way	LRU	0	24	0.00%
11	128 B	2 B	2-way	LRU	0	24	0.00%
12	128 B	2 B	4-way	LRU	0	24	0.00%
13	128 B	4 B	1-way	LRU	0	24	0.00%
14	128 B	4 B	2-way	LRU	0	24	0.00%
15	128 B	4 B	4-way	LRU	0	24	0.00%

16	128 B	8 B	1-way	LRU	12	12	50.00%
17	128 B	8 B	2-way	LRU	12	12	50.00%
18	128 B	8 B	4-way	LRU	12	12	50.00%
19	256 B	2 B	1-way	LRU	0	24	0.00%
20	256 B	2 B	2-way	LRU	0	24	0.00%
21	256 B	2 B	4-way	LRU	0	24	0.00%
22	256 B	4 B	1-way	LRU	0	24	0.00%
23	256 B	4 B	2-way	LRU	0	24	0.00%
24	256 B	4 B	4-way	LRU	0	24	0.00%
25	256 B	8 B	1-way	LRU	12	12	50.00%
26	256 B	8 B	2-way	LRU	12	12	50.00%
27	256 B	8 B	4-way	LRU	12	12	50.00%
28	64 B	2 B	1-way	FIFO	0	24	0.00%
29	64 B	2 B	2-way	FIFO	0	24	0.00%
30	64 B	2 B	4-way	FIFO	0	24	0.00%
31	64 B	4 B	1-way	FIFO	0	24	0.00%
32	64 B	4 B	2-way	FIFO	0	24	0.00%
33	64 B	4 B	4-way	FIFO	0	24	0.00%
34	64 B	8 B	1-way	FIFO	12	12	50.00%
35	64 B	8 B	2-way	FIFO	12	12	50.00%
36	64 B	8 B	4-way	FIFO	12	12	50.00%
37	128 B	2 B	1-way	FIFO	0	24	0.00%
38	128 B	2 B	2-way	FIFO	0	24	0.00%
39	128 B	2 B	4-way	FIFO	0	24	0.00%
40	128 B	4 B	1-way	FIFO	0	24	0.00%
41	128 B	4 B	2-way	FIFO	0	24	0.00%
42	128 B	4 B	4-way	FIFO	0	24	0.00%
43	128 B	8 B	1-way	FIFO	12	12	50.00%
44	128 B	8 B	2-way	FIFO	12	12	50.00%
45	128 B	8 B	4-way	FIFO	12	12	50.00%
46	256 B	2 B	1-way	FIFO	0	24	0.00%
47	256 B	2 B	2-way	FIFO	0	24	0.00%
48	256 B	2 B	4-way	FIFO	0	24	0.00%
49	256 B	4 B	1-way	FIFO	0	24	0.00%

50	256 B	4 B	2-way	FIFO	0	24	0.00%
51	256 B	4 B	4-way	FIFO	0	24	0.00%
52	256 B	8 B	1-way	FIFO	12	12	50.00%
53	256 B	8 B	2-way	FIFO	12	12	50.00%
54	256 B	8 B	4-way	FIFO	12	12	50.00%
55	64 B	2 B	1-way	Random	0	24	0.00%
56	64 B	2 B	2-way	Random	0	24	0.00%
57	64 B	2 B	4-way	Random	0	24	0.00%
58	64 B	4 B	1-way	Random	0	24	0.00%
59	64 B	4 B	2-way	Random	0	24	0.00%
60	64 B	4 B	4-way	Random	0	24	0.00%
61	64 B	8 B	1-way	Random	12	12	50.00%
62	64 B	8 B	2-way	Random	12	12	50.00%
63	64 B	8 B	4-way	Random	12	12	50.00%
64	128 B	2 B	1-way	Random	0	24	0.00%
65	128 B	2 B	2-way	Random	0	24	0.00%
66	128 B	2 B	4-way	Random	0	24	0.00%
67	128 B	4 B	1-way	Random	0	24	0.00%
68	128 B	4 B	2-way	Random	0	24	0.00%
69	128 B	4 B	4-way	Random	0	24	0.00%
70	128 B	8 B	1-way	Random	12	12	50.00%
71	128 B	8 B	2-way	Random	12	12	50.00%
72	128 B	8 B	4-way	Random	12	12	50.00%
73	256 B	2 B	1-way	Random	0	24	0.00%
74	256 B	2 B	2-way	Random	0	24	0.00%
75	256 B	2 B	4-way	Random	0	24	0.00%
76	256 B	4 B	1-way	Random	0	24	0.00%
77	256 B	4 B	2-way	Random	0	24	0.00%
78	256 B	4 B	4-way	Random	0	24	0.00%
79	256 B	8 B	1-way	Random	12	12	50.00%
80	256 B	8 B	2-way	Random	12	12	50.00%
81	256 B	8 B	4-way	Random	12	12	50.00%

APPENDIX E: Full 81-Run Replacement Policy Sweep for Trace 5

Test No	Cache Size (C)	Block Size (B)	Associativity (E)	Policy	Hits	Misses	Hit Rate
1	64 B	2 B	1-way (Direct)	LRU	0	24	0.00%
2	64 B	2 B	2-way	LRU	16	8	66.67%
3	64 B	2 B	4-way	LRU	16	8	66.67%
4	64 B	4 B	1-way	LRU	0	24	0.00%
5	64 B	4 B	2-way	LRU	16	8	66.67%
6	64 B	4 B	4-way	LRU	16	8	66.67%
7	64 B	8 B	1-way	LRU	12	12	50.00%
8	64 B	8 B	2-way	LRU	20	4	83.33%
9	64 B	8 B	4-way	LRU	20	4	83.33%
10	128 B	2 B	1-way	LRU	0	24	0.00%
11	128 B	2 B	2-way	LRU	16	8	66.67%
12	128 B	2 B	4-way	LRU	16	8	66.67%
13	128 B	4 B	1-way	LRU	0	24	0.00%
14	128 B	4 B	2-way	LRU	16	8	66.67%
15	128 B	4 B	4-way	LRU	16	8	66.67%
16	128 B	8 B	1-way	LRU	12	12	50.00%
17	128 B	8 B	2-way	LRU	20	4	83.33%
18	128 B	8 B	4-way	LRU	20	4	83.33%
19	256 B	2 B	1-way	LRU	0	24	0.00%
20	256 B	2 B	2-way	LRU	16	8	66.67%
21	256 B	2 B	4-way	LRU	16	8	66.67%
22	256 B	4 B	1-way	LRU	0	24	0.00%
23	256 B	4 B	2-way	LRU	16	8	66.67%
24	256 B	4 B	4-way	LRU	16	8	66.67%
25	256 B	8 B	1-way	LRU	12	12	50.00%
26	256 B	8 B	2-way	LRU	20	4	83.33%
27	256 B	8 B	4-way	LRU	20	4	83.33%
28	64 B	2 B	1-way	FIFO	0	24	0.00%
29	64 B	2 B	2-way	FIFO	16	8	66.67%
30	64 B	2 B	4-way	FIFO	16	8	66.67%
31	64 B	4 B	1-way	FIFO	0	24	0.00%
32	64 B	4 B	2-way	FIFO	16	8	66.67%

33	64 B	4 B	4-way	FIFO	16	8	66.67%
34	64 B	8 B	1-way	FIFO	12	12	50.00%
35	64 B	8 B	2-way	FIFO	20	4	83.33%
36	64 B	8 B	4-way	FIFO	20	4	83.33%
37	128 B	2 B	1-way	FIFO	0	24	0.00%
38	128 B	2 B	2-way	FIFO	16	8	66.67%
39	128 B	2 B	4-way	FIFO	16	8	66.67%
40	128 B	4 B	1-way	FIFO	0	24	0.00%
41	128 B	4 B	2-way	FIFO	16	8	66.67%
42	128 B	4 B	4-way	FIFO	16	8	66.67%
43	128 B	8 B	1-way	FIFO	12	12	50.00%
44	128 B	8 B	2-way	FIFO	20	4	83.33%
45	128 B	8 B	4-way	FIFO	20	4	83.33%
46	256 B	2 B	1-way	FIFO	0	24	0.00%
47	256 B	2 B	2-way	FIFO	16	8	66.67%
48	256 B	2 B	4-way	FIFO	16	8	66.67%
49	256 B	4 B	1-way	FIFO	0	24	0.00%
50	256 B	4 B	2-way	FIFO	16	8	66.67%
51	256 B	4 B	4-way	FIFO	16	8	66.67%
52	256 B	8 B	1-way	FIFO	12	12	50.00%
53	256 B	8 B	2-way	FIFO	20	4	83.33%
54	256 B	8 B	4-way	FIFO	20	4	83.33%
55	64 B	2 B	1-way	Random	0	24	0.00%
56	64 B	2 B	2-way	Random	16	8	66.67%
57	64 B	2 B	4-way	Random	16	8	66.67%
58	64 B	4 B	1-way	Random	0	24	0.00%
59	64 B	4 B	2-way	Random	16	8	66.67%
60	64 B	4 B	4-way	Random	16	8	66.67%
61	64 B	8 B	1-way	Random	12	12	50.00%
62	64 B	8 B	2-way	Random	20	4	83.33%
63	64 B	8 B	4-way	Random	20	4	83.33%
64	128 B	2 B	1-way	Random	0	24	0.00%
65	128 B	2 B	2-way	Random	16	8	66.67%
66	128 B	2 B	4-way	Random	16	8	66.67%

67	128 B	4 B	1-way	Random	0	24	0.00%
68	128 B	4 B	2-way	Random	16	8	66.67%
69	128 B	4 B	4-way	Random	16	8	66.67%
70	128 B	8 B	1-way	Random	12	12	50.00%
71	128 B	8 B	2-way	Random	20	4	83.33%
72	128 B	8 B	4-way	Random	20	4	83.33%
73	256 B	2 B	1-way	Random	0	24	0.00%
74	256 B	2 B	2-way	Random	16	8	66.67%
75	256 B	2 B	4-way	Random	16	8	66.67%
76	256 B	4 B	1-way	Random	0	24	0.00%
77	256 B	4 B	2-way	Random	16	8	66.67%
78	256 B	4 B	4-way	Random	16	8	66.67%
79	256 B	8 B	1-way	Random	12	12	50.00%
80	256 B	8 B	2-way	Random	20	4	83.33%
81	256 B	8 B	4-way	Random	20	4	83.33%