Cache Memory Simulator: Documentation and Analysis

Introduction

The Cache Memory Simulator is a web-based tool designed to emulate the behavior of a Block-Set Associative Cache with the Most Recently Used (MRU) replacement policy. This tool provides an interactive interface for users to configure cache parameters, run simulations, and analyze the results to better understand cache memory performance.

Features

- **Flexible Configuration**: Users can set block size, number of sets, cache size, and cycle times.
- **Simulation of Program Flow**: Users can input a sequence of memory accesses and observe cache behavior.
- **Comprehensive Results**: The simulator provides detailed metrics including cache hits, misses, miss penalty, average memory access time, and total memory access time.
- **Cache Memory Snapshot**: Visualize the state of the cache memory after simulation.
- **Export Results**: Save simulation results to a text file for further analysis or record-keeping.

User Interface

Form Inputs

1. **Block Size**: Defines the size of each block in the cache.
2. **Set Size**: Specifies the number of blocks per set in the cache.
3. **Main Memory Size**: Indicates the total size of the main memory, with units in either blocks or words.
4. **Cache Memory Size**: Defines the total size of the cache, with units in either blocks or words.
5. **Cache Cycle Time**: The time taken to access data from the cache, in nanoseconds.
6. **Memory Cycle Time**: The time taken to access data from the main memory, in nanoseconds.
7. **Program Flow**: A sequence of memory accesses to simulate, specified in either blocks or words.

Output Results

1. **Cache Hits**: Number of times the requested data was found in the cache.
2. **Cache Misses**: Number of times the requested data was not found in the cache.
3. **Miss Penalty**: The time penalty incurred for a cache miss.
4. **Average Memory Access Time**: The average time to access data from the cache or main memory.
5. **Total Memory Access Time**: The total time spent accessing memory during the simulation.
6. **Cache Snapshot**: A textual representation of the cache state after simulation.
7. **Generated Text**: A formatted summary of the simulation results for download.

Simulation Logic

Cache Configuration

1. **Block Size and Set Size**: The block size determines the amount of data transferred per cache line, while the set size determines how many blocks can be stored in each set. The cache memory is divided into sets, and each set can hold a number of blocks equal to the set size.
2. **Conversion of Memory Sizes**: If the memory size or cache size is specified in words, it is converted to blocks based on the block size. This ensures that all calculations are consistent.

Cache Replacement Policy

1. **Most Recently Used (MRU)**: When a block needs to be replaced in the cache, the MRU policy evicts the block that was accessed most recently. This ensures that frequently accessed data remains in the cache for longer periods.

Simulation Steps

1. **Input Validation**: The program checks if the input values are valid and within the acceptable range. This includes verifying that the program flow does not exceed the main memory size and that the sizes are correctly divisible.
2. **Cache Operations**: For each memory access in the program flow:
    - The cache is checked for the requested block.
    - If a cache hit occurs, the block's timestamp is updated.
    - If a cache miss occurs, a replacement is performed according to the MRU policy.
3. **Metrics Calculation**:
    - **Hit Rate**: The ratio of cache hits to the total number of accesses.

    - **Miss Penalty**: Calculated as twice the cache cycle time plus the product of block size and memory cycle time.
    - **Average Access Time**: Weighted average of cache access time and miss penalty.
    - **Total Access Time**: Total time spent on cache hits and misses.

Error Handling

1. **Input Errors**: Errors in input values, such as invalid memory addresses or incompatible block sizes, are caught and displayed to the user.
2. **Simulation Errors**: Errors during the simulation, such as exceeding memory size or invalid program flow, are handled gracefully with appropriate error messages.

Analysis

Performance Metrics

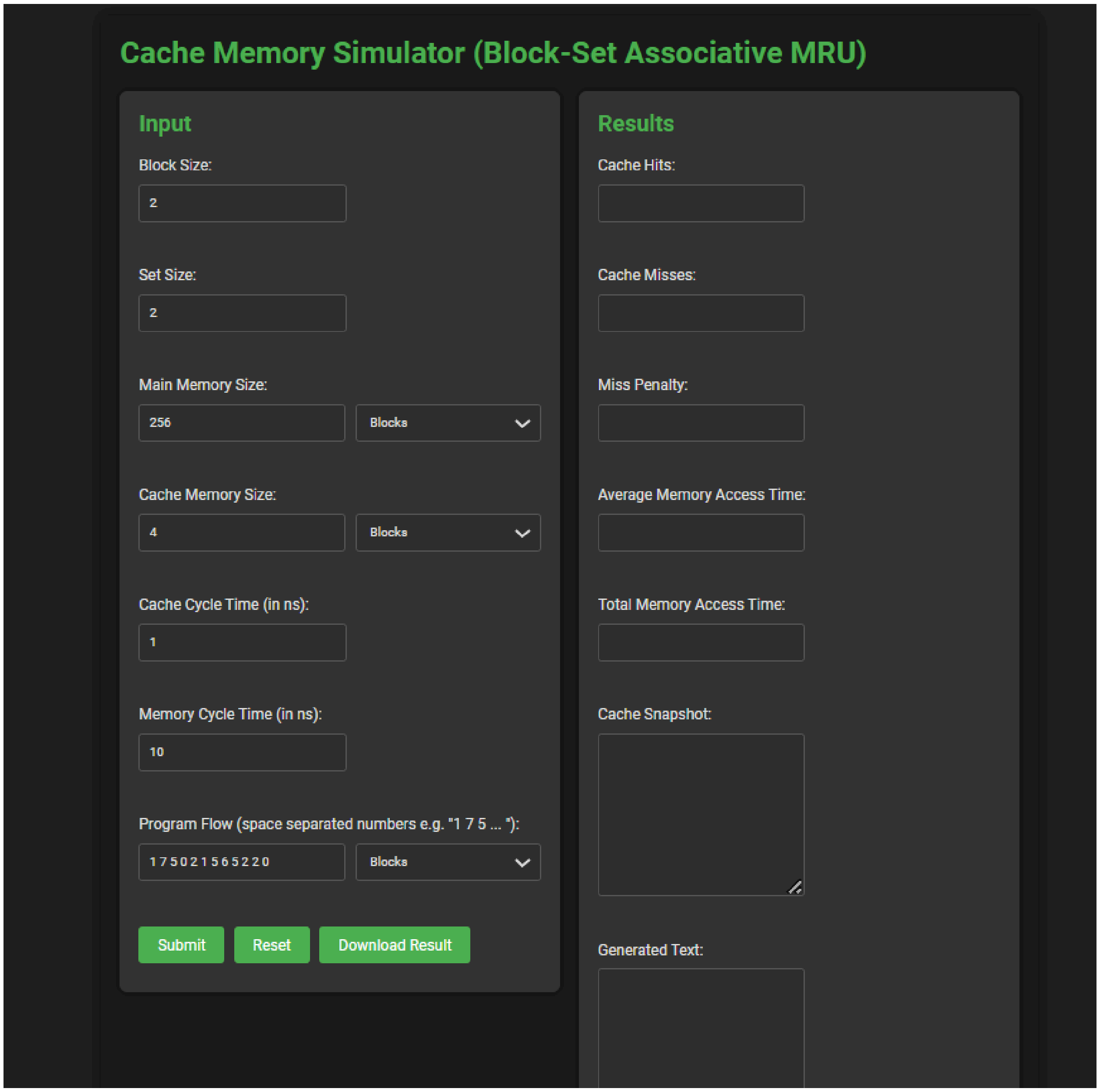
- **Cache Hits and Misses**: Understanding the ratio of cache hits to misses provides insight into the effectiveness of the cache configuration. A higher hit rate indicates better cache performance.
- **Miss Penalty**: A higher miss penalty can significantly impact the overall performance, especially if the memory cycle time is high.
- **Average Memory Access Time**: This metric combines the effects of both cache hits and misses, giving a comprehensive view of memory access performance.

Visualization

- **Cache Snapshot**: Provides a clear view of the cache state, helping users understand which blocks are currently stored and how they are organized.

Usability

The user interface is designed to be intuitive, with clearly labeled inputs and outputs. The form-based input system allows users to easily configure parameters and run simulations. The results are presented in a user-friendly format, making it easy to analyze and interpret the simulation outcomes.



Cache Memory Simulator (Block-Set Associative MRU)

Input

Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 7 5 0 2 1 5 6 5 2 2 0 Blocks

Results

Cache Hits:

Cache Misses:

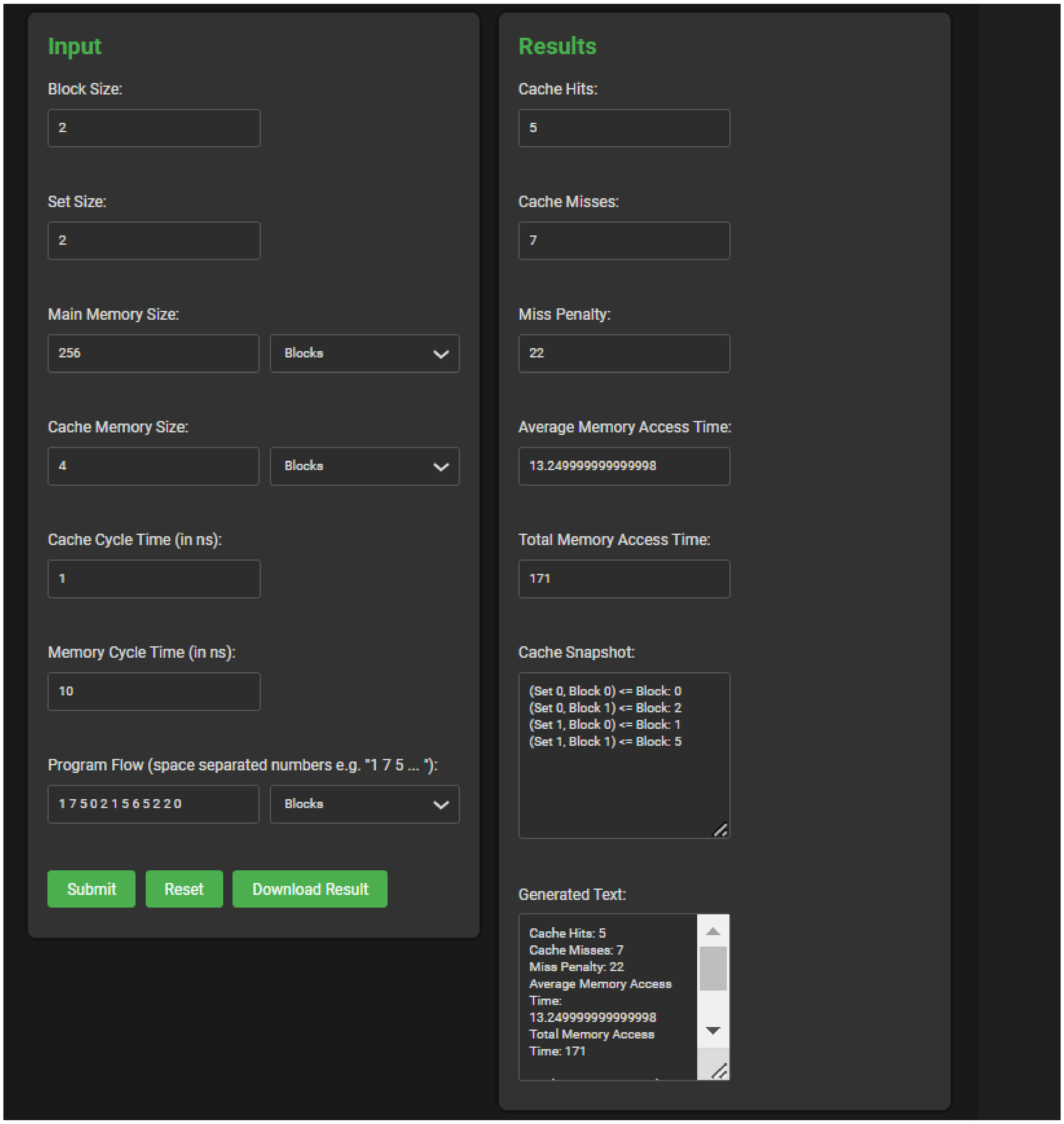
Miss Penalty:

Average Memory Access Time:

Total Memory Access Time:

Cache Snapshot:

Generated Text:



Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 7 5 0 2 1 5 6 5 2 2 0 Blocks

Cache Hits: 5

Cache Misses: 7

Miss Penalty: 22

Average Memory Access Time: 13.249999999999998

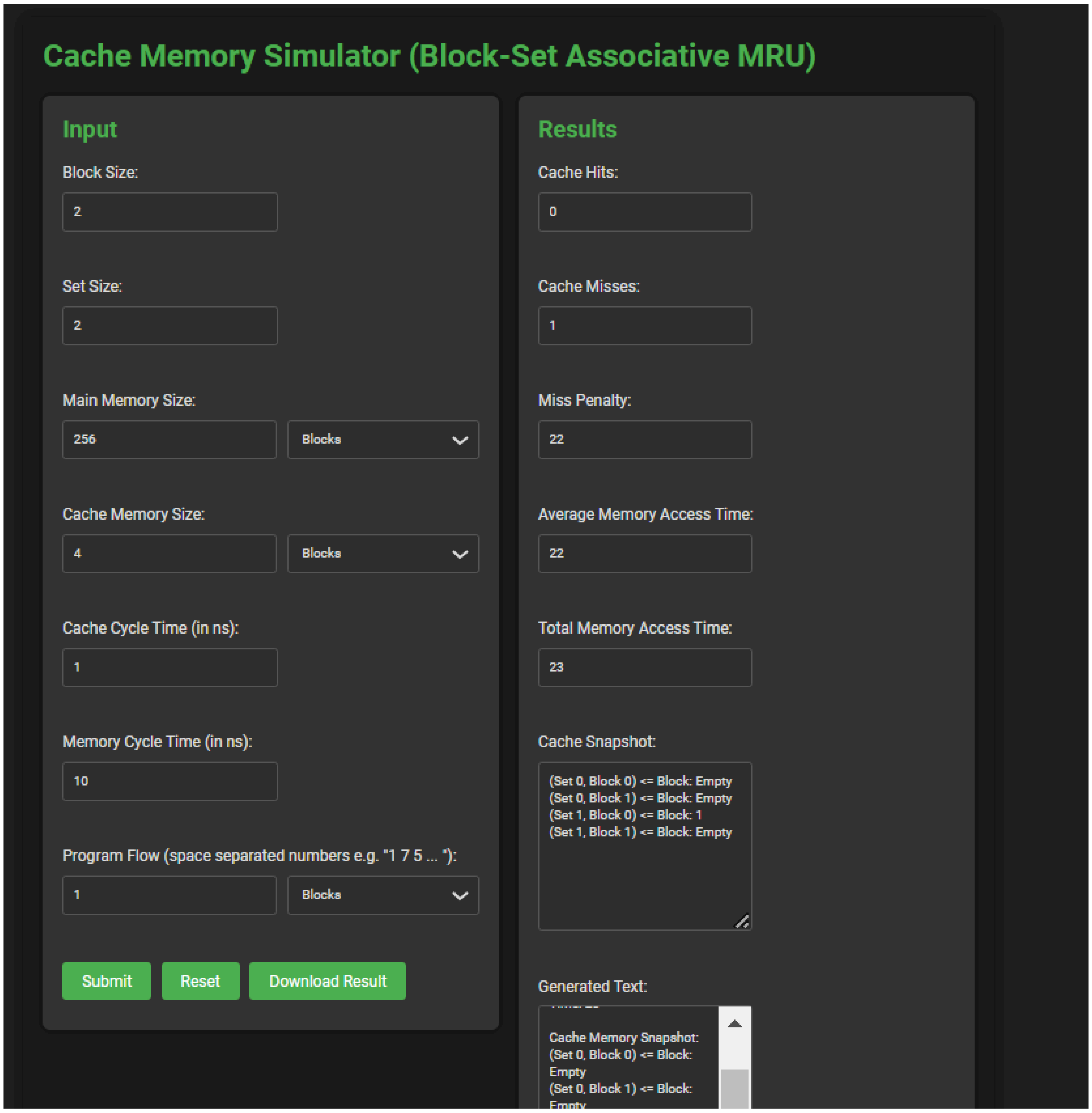
Total Memory Access Time: 171

Cache Snapshot:

```
(Set 0, Block 0) <= Block: 0
(Set 0, Block 1) <= Block: 2
(Set 1, Block 0) <= Block: 1
(Set 1, Block 1) <= Block: 5
```

Generated Text:

```
Cache Hits: 5
Cache Misses: 7
Miss Penalty: 22
Average Memory Access
Time:
13.249999999999998
Total Memory Access
Time: 171
```


Cache Memory Simulator (Block-Set Associative MRU)

Input

Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 Blocks

Results

Cache Hits: 0

Cache Misses: 1

Miss Penalty: 22

Average Memory Access Time: 22

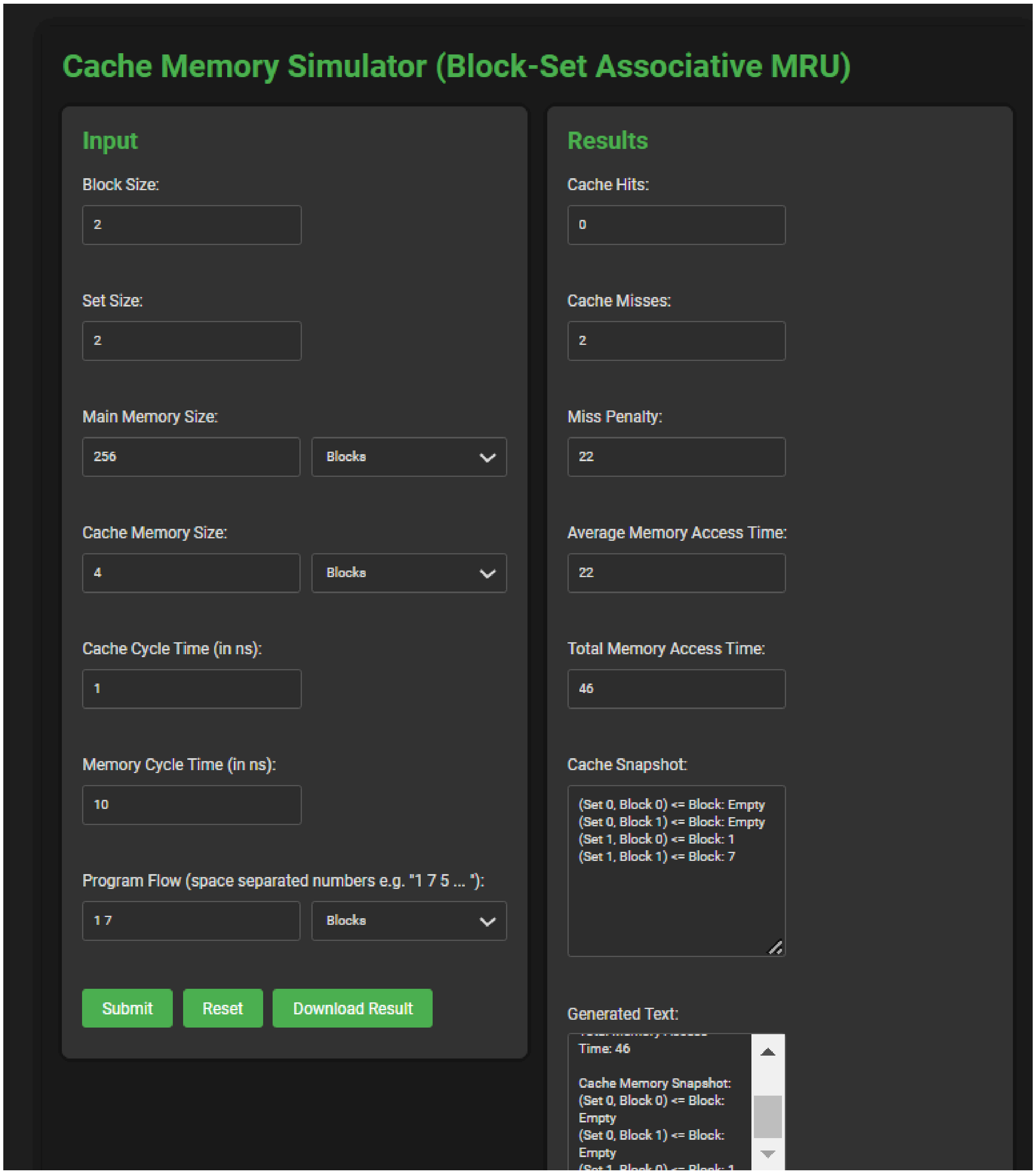
Total Memory Access Time: 23

Cache Snapshot:

```
(Set 0, Block 0) <= Block: Empty
(Set 0, Block 1) <= Block: Empty
(Set 1, Block 0) <= Block: 1
(Set 1, Block 1) <= Block: Empty
```

Generated Text:

```
Cache Memory Snapshot:
(Set 0, Block 0) <= Block:
Empty
(Set 0, Block 1) <= Block:
Empty
```



Cache Memory Simulator (Block-Set Associative MRU)

Input

Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 7 Blocks

Results

Cache Hits: 0

Cache Misses: 2

Miss Penalty: 22

Average Memory Access Time: 22

Total Memory Access Time: 46

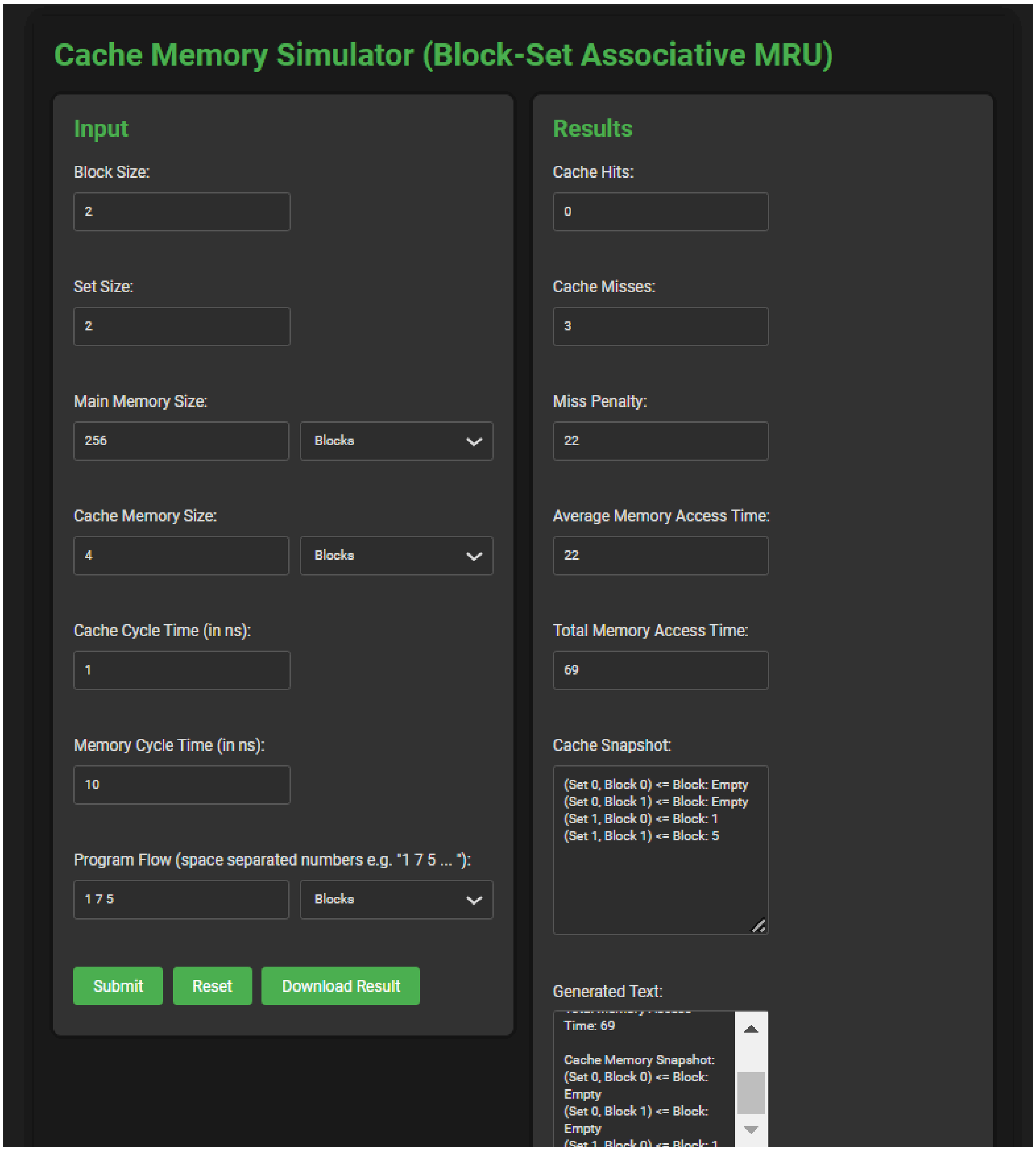
Cache Snapshot:

```
(Set 0, Block 0) <= Block: Empty
(Set 0, Block 1) <= Block: Empty
(Set 1, Block 0) <= Block: 1
(Set 1, Block 1) <= Block: 7
```

Generated Text:

```
Time: 46

Cache Memory Snapshot:
(Set 0, Block 0) <= Block:
Empty
(Set 0, Block 1) <= Block:
Empty
```



Cache Memory Simulator (Block-Set Associative MRU)

Input

Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 7 5 Blocks

Results

Cache Hits: 0

Cache Misses: 3

Miss Penalty: 22

Average Memory Access Time: 22

Total Memory Access Time: 69

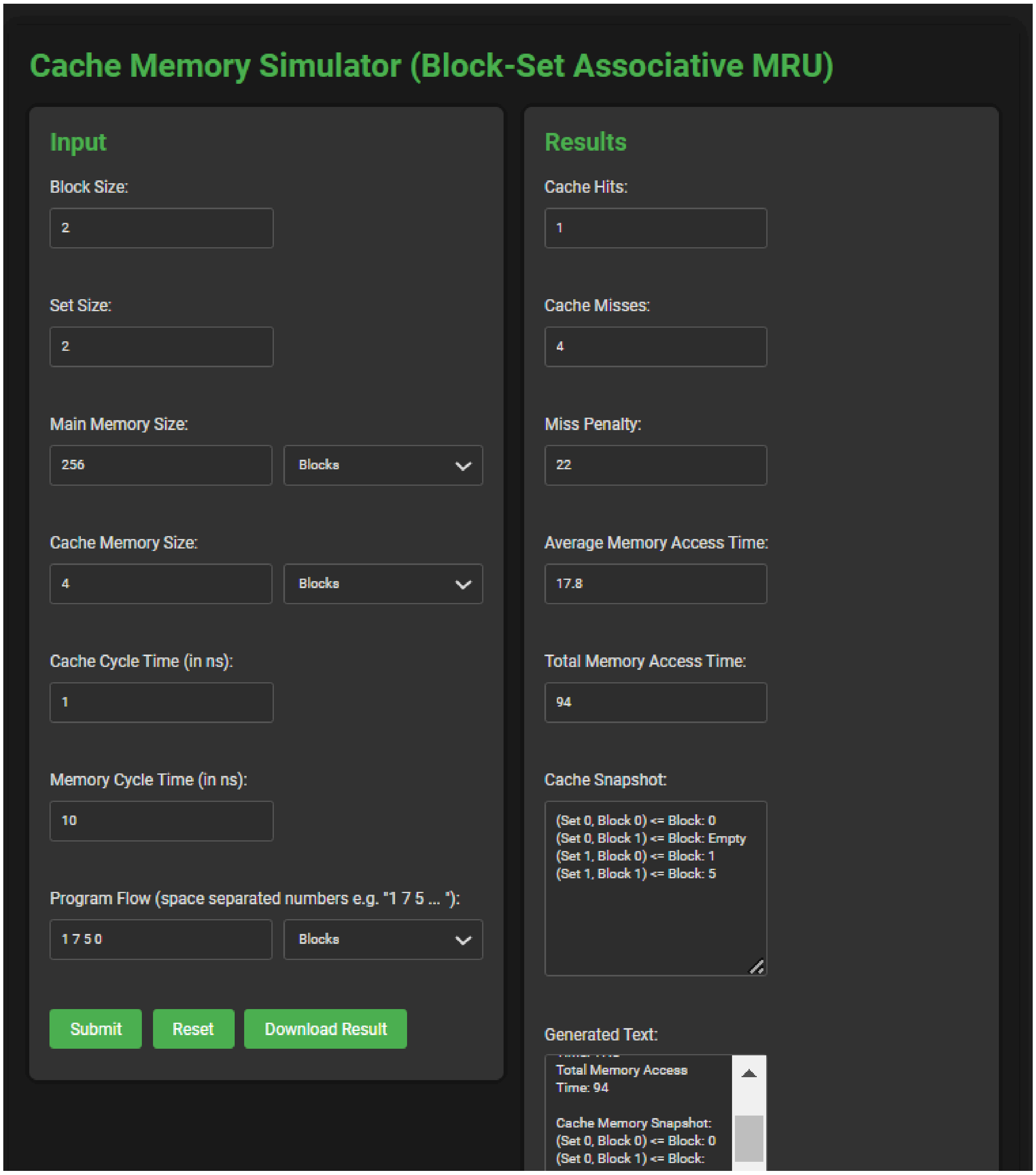
Cache Snapshot:

```
(Set 0, Block 0) <= Block: Empty
(Set 0, Block 1) <= Block: Empty
(Set 1, Block 0) <= Block: 1
(Set 1, Block 1) <= Block: 5
```

Generated Text:

```
Total Memory Access
Time: 69

Cache Memory Snapshot:
(Set 0, Block 0) <= Block:
Empty
(Set 0, Block 1) <= Block:
Empty
(Set 1, Block 0) <= Block: 1
```



Cache Memory Simulator (Block-Set Associative MRU)

Input

Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 7 5 0 Blocks

Results

Cache Hits: 1

Cache Misses: 4

Miss Penalty: 22

Average Memory Access Time: 17.8

Total Memory Access Time: 94

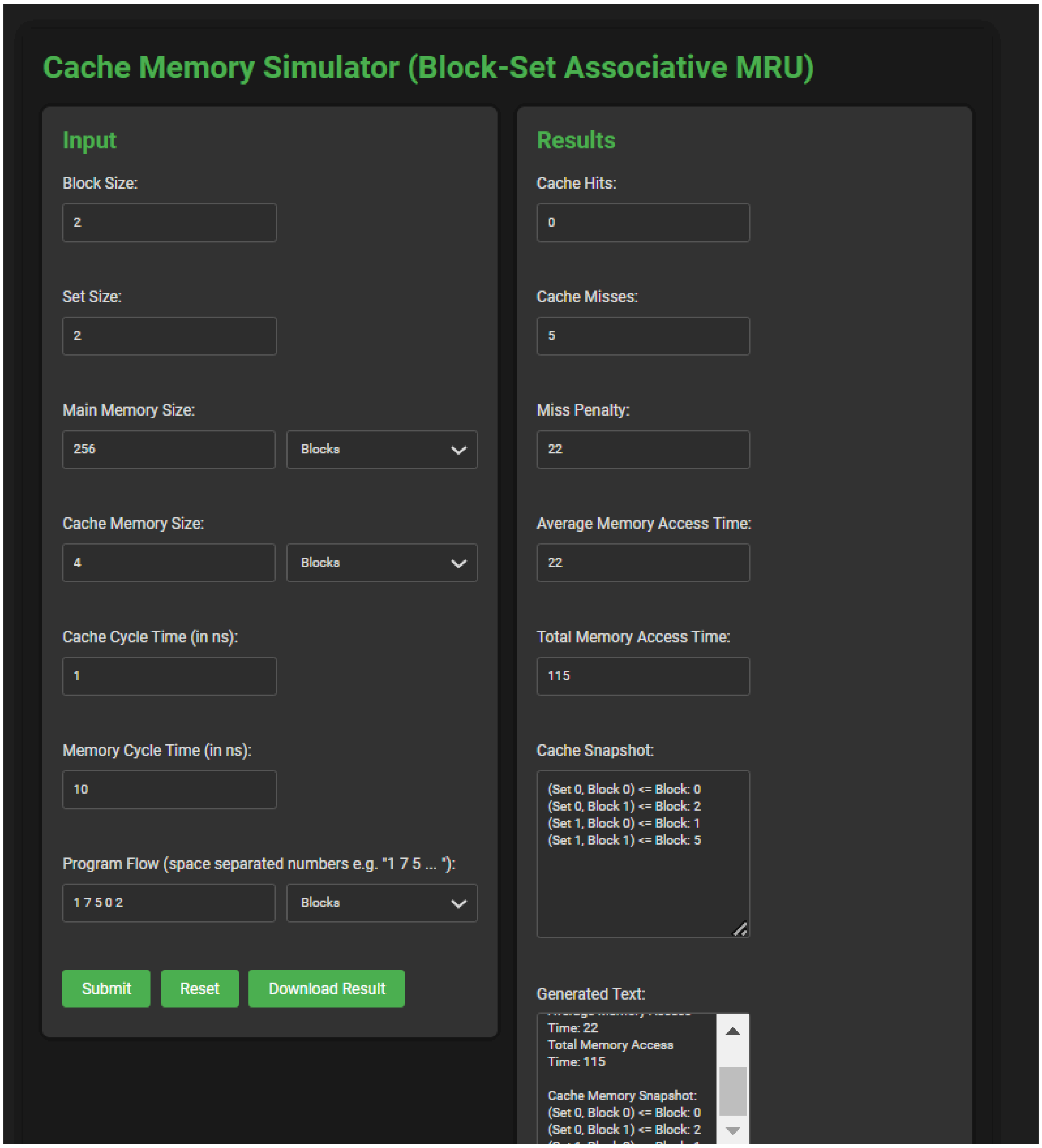
Cache Snapshot:

```
(Set 0, Block 0) <= Block: 0
(Set 0, Block 1) <= Block: Empty
(Set 1, Block 0) <= Block: 1
(Set 1, Block 1) <= Block: 5
```

Generated Text:

```
Total Memory Access
Time: 94

Cache Memory Snapshot:
(Set 0, Block 0) <= Block: 0
(Set 0, Block 1) <= Block:
```



Cache Memory Simulator (Block-Set Associative MRU)

Input

Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 7 5 0 2 Blocks

Results

Cache Hits: 0

Cache Misses: 5

Miss Penalty: 22

Average Memory Access Time: 22

Total Memory Access Time: 115

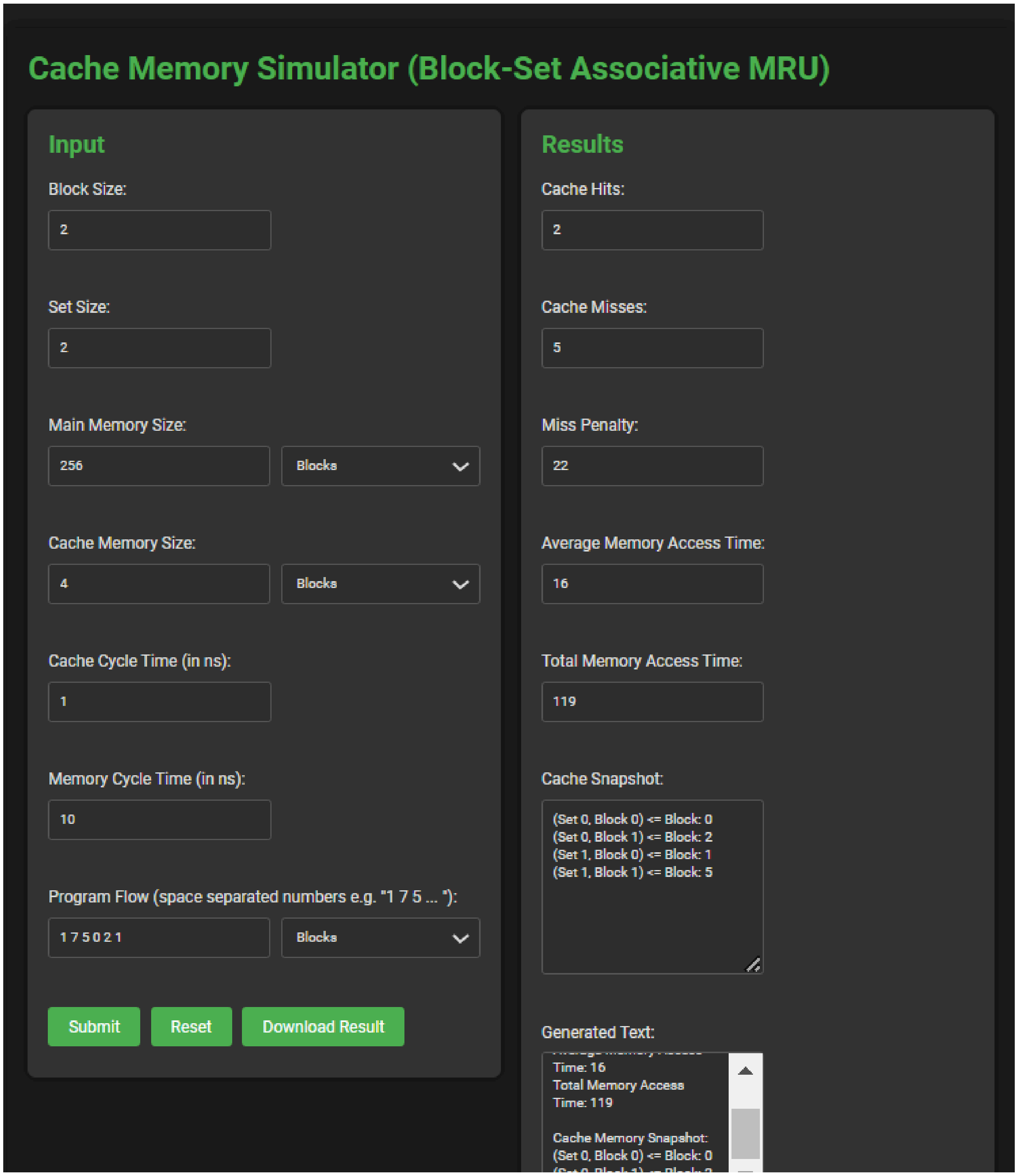
Cache Snapshot:

```
(Set 0, Block 0) <= Block: 0
(Set 0, Block 1) <= Block: 2
(Set 1, Block 0) <= Block: 1
(Set 1, Block 1) <= Block: 5
```

Generated Text:

```
Time: 22
Total Memory Access
Time: 115

Cache Memory Snapshot:
(Set 0, Block 0) <= Block: 0
(Set 0, Block 1) <= Block: 2
```



Cache Memory Simulator (Block-Set Associative MRU)

Input

Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 7 5 0 2 1 Blocks

Results

Cache Hits: 2

Cache Misses: 5

Miss Penalty: 22

Average Memory Access Time: 16

Total Memory Access Time: 119

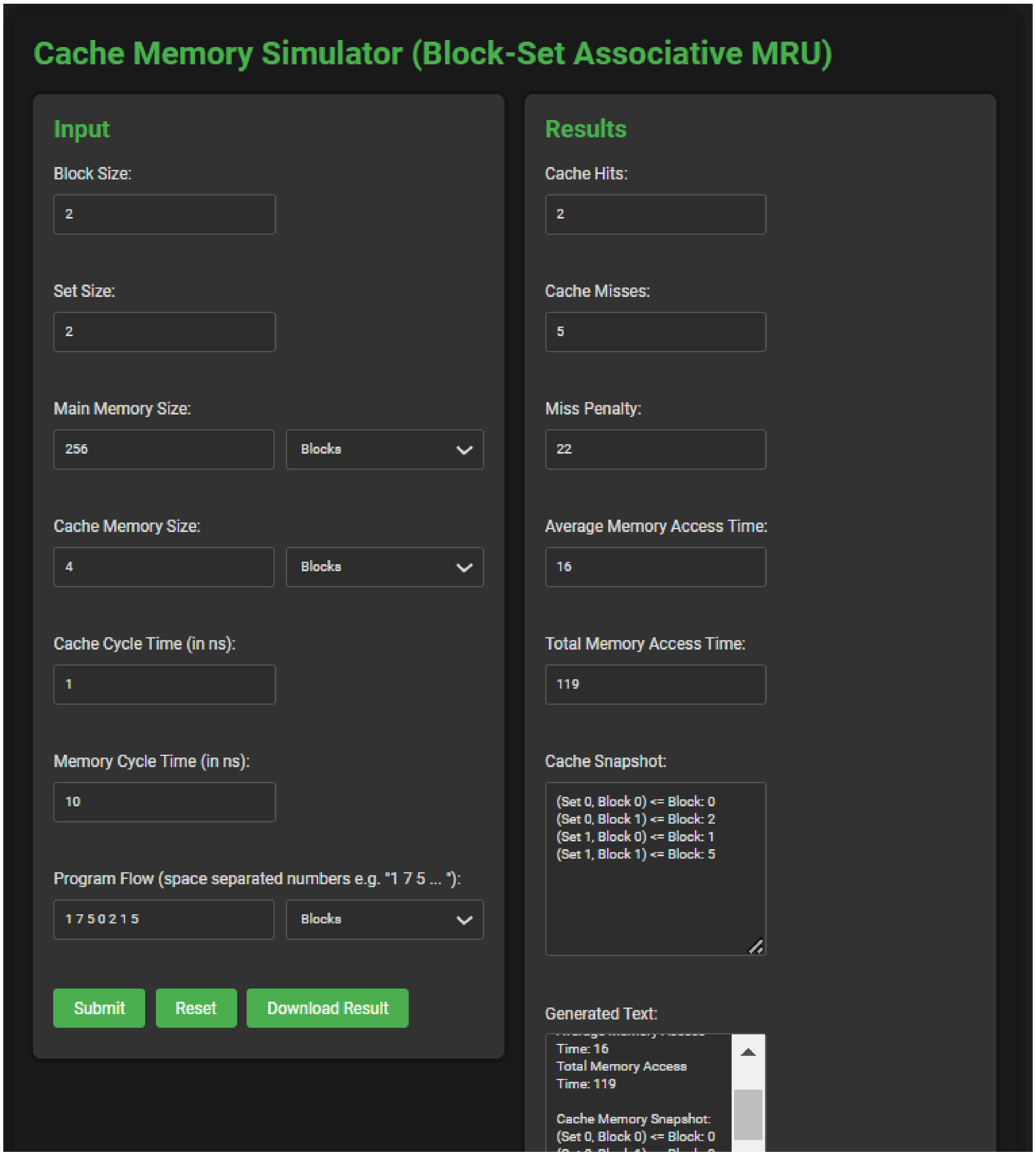
Cache Snapshot:

```
(Set 0, Block 0) <= Block: 0
(Set 0, Block 1) <= Block: 2
(Set 1, Block 0) <= Block: 1
(Set 1, Block 1) <= Block: 5
```

Generated Text:

```
Time: 16
Total Memory Access
Time: 119

Cache Memory Snapshot:
(Set 0, Block 0) <= Block: 0
```



Cache Memory Simulator (Block-Set Associative MRU)

Input

Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 7 5 0 2 1 5 Blocks

Results

Cache Hits: 2

Cache Misses: 5

Miss Penalty: 22

Average Memory Access Time: 16

Total Memory Access Time: 119

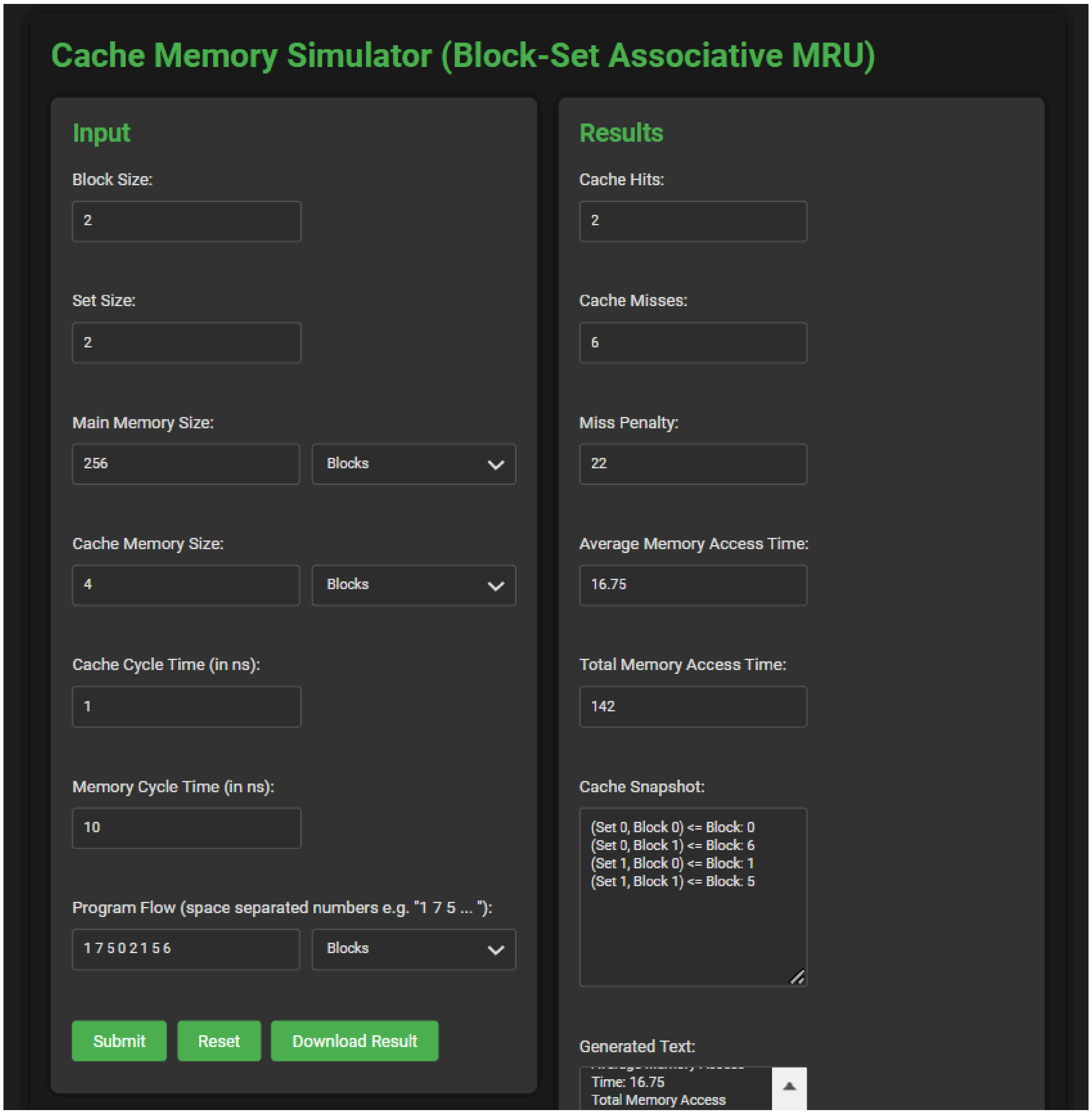
Cache Snapshot:

```
(Set 0, Block 0) <= Block: 0
(Set 0, Block 1) <= Block: 2
(Set 1, Block 0) <= Block: 1
(Set 1, Block 1) <= Block: 5
```

Generated Text:

```
Time: 16
Total Memory Access
Time: 119

Cache Memory Snapshot:
(Set 0, Block 0) <= Block: 0
```



Cache Memory Simulator (Block-Set Associative MRU)

Input

Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 7 5 0 2 1 5 6 Blocks

Results

Cache Hits: 2

Cache Misses: 6

Miss Penalty: 22

Average Memory Access Time: 16.75

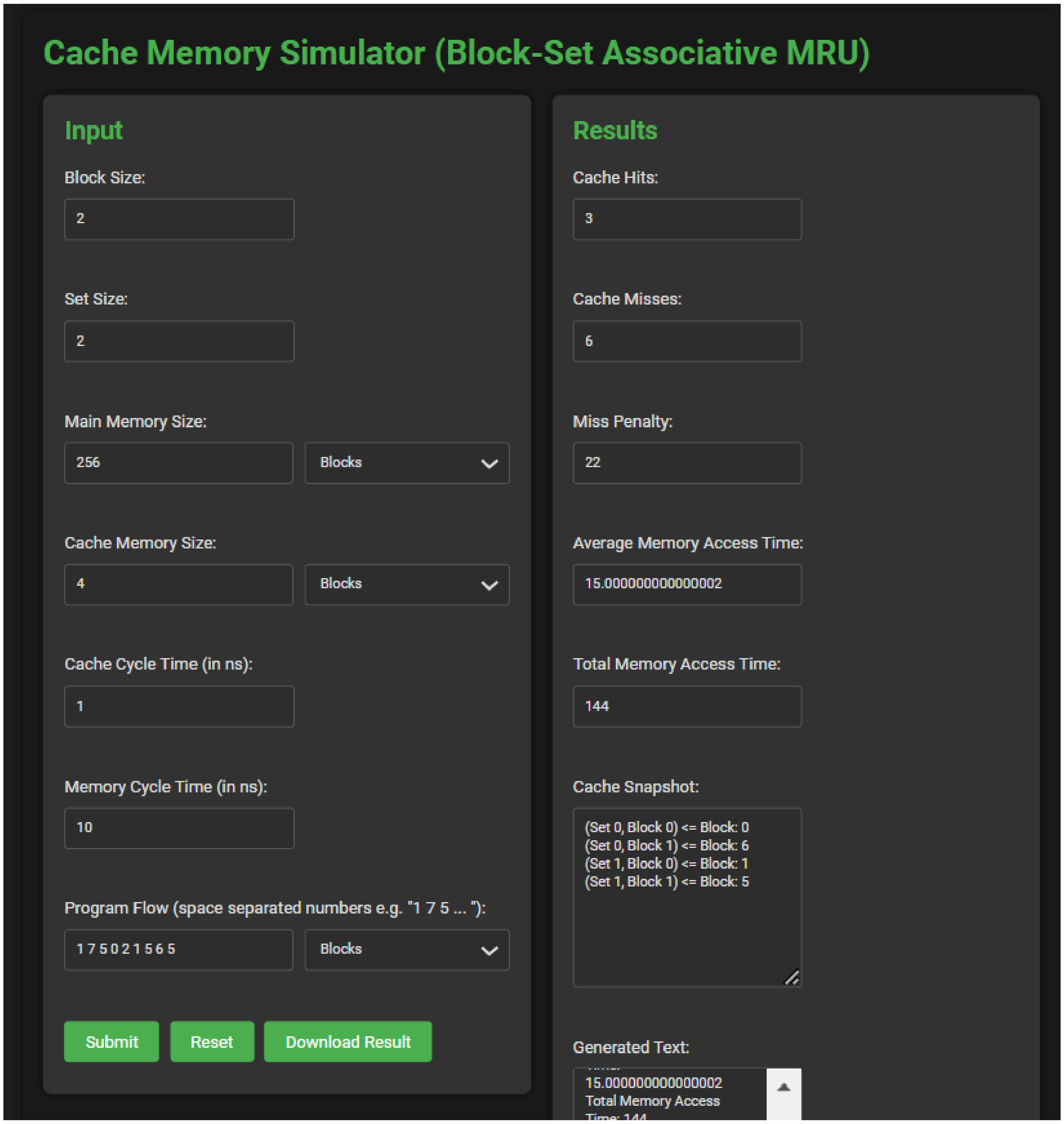
Total Memory Access Time: 142

Cache Snapshot:

```
(Set 0, Block 0) <= Block: 0
(Set 0, Block 1) <= Block: 6
(Set 1, Block 0) <= Block: 1
(Set 1, Block 1) <= Block: 5
```

Generated Text:

```
Time: 16.75
Total Memory Access
```



Cache Memory Simulator (Block-Set Associative MRU)

Input

Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 7 5 0 2 1 5 6 5 Blocks

Results

Cache Hits: 3

Cache Misses: 6

Miss Penalty: 22

Average Memory Access Time: 15.000000000000002

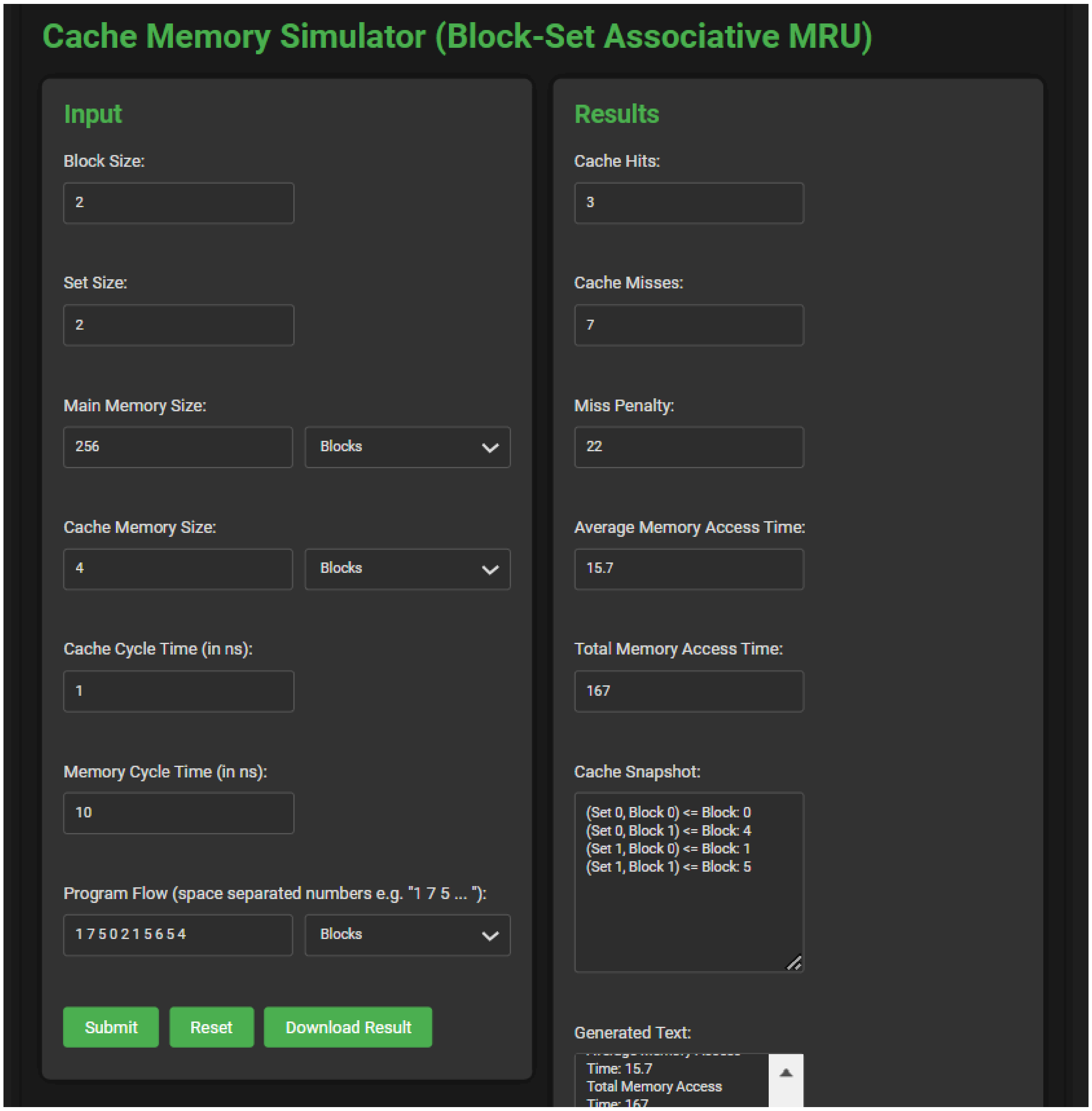
Total Memory Access Time: 144

Cache Snapshot:

```
(Set 0, Block 0) <= Block: 0
(Set 0, Block 1) <= Block: 6
(Set 1, Block 0) <= Block: 1
(Set 1, Block 1) <= Block: 5
```

Generated Text:

```
15.000000000000002
Total Memory Access
Time: 144
```



Cache Memory Simulator (Block-Set Associative MRU)

Input

Block Size: 2

Set Size: 2

Main Memory Size: 256 Blocks

Cache Memory Size: 4 Blocks

Cache Cycle Time (in ns): 1

Memory Cycle Time (in ns): 10

Program Flow (space separated numbers e.g. "1 7 5 ... "): 1 7 5 0 2 1 5 6 5 4 Blocks

Results

Cache Hits: 3

Cache Misses: 7

Miss Penalty: 22

Average Memory Access Time: 15.7

Total Memory Access Time: 167

Cache Snapshot:

```
(Set 0, Block 0) <= Block: 0
(Set 0, Block 1) <= Block: 4
(Set 1, Block 0) <= Block: 1
(Set 1, Block 1) <= Block: 5
```

Generated Text:

```
Time: 15.7
Total Memory Access
```

Conclusion

The Cache Memory Simulator provides a robust tool for understanding and analyzing cache memory behavior. By allowing users to configure various parameters and visualize simulation results, it offers valuable insights into cache performance and memory access patterns.