Cache Simulator for Exploring Performance Metrics of m-way Set-Associative

EE 275 Advanced Computer Architecture

Final Project Submission

By:

Kamlesh Kumar
Student ID- 017047212
November 27, 2024

TABLE OF CONTENTS

ABSTRACT

This project focuses on developing an advanced cache simulator to analyze the performance of m-way set-associative caches across various configurations, ranging from simple direct-mapped designs to fully associative ones. The simulator will evaluate performance metrics such as hit ratio and miss ratio using Verilog and the Least Recently Used (LRU) replacement algorithm. By experimenting with different cache sizes, line sizes, and associativities, we aim to understand their impact on cache performance and identify key factors that enhance efficiency.

Through detailed simulations and analyses, this study seeks to optimize cache design by balancing hit rate improvement with minimal storage costs. Line size, which influences hit ratio based on block usage frequency, and higher associativity, which reduces miss rates, are critical parameters in this optimization. Ultimately, the objective is to provide insights into cache operation and configuration, equipping system designers with the knowledge needed to make informed decisions and enhance overall system performance.

INTRODUCTION

The primary goal of this project is to develop a highly accurate cache simulator. Cache memory, a critical component of modern computer architecture, significantly reduces memory access time compared to traditional hard drives. By enabling faster access to frequently used data, cache memory plays a vital role in enhancing system performance. In this project, we present a Verilog-based cache simulator tailored for m-way set-associative caches, where m can range from 1 (direct-mapped) to fully associative configurations, supporting values such as 2, 4, 8, 16, and 32. The simulator accommodates cache sizes up to 128 KB, making it suitable for large-scale data processing.

The simulator incorporates the Least Recently Used (LRU) algorithm for cache replacement, ensuring efficient management of cache blocks. It focuses on three key design parameters: block (or line) size, cache size, and set associativity. To track the sequence of memory load instructions executed by programs, the simulator uses an address trace file, where each entry logs a byte address. To optimize storage, the trace records the difference between consecutive addresses rather than storing absolute addresses. The project is designed exclusively for byte-addressable memory, ensuring precision and relevance in its simulations.

Accompanying the simulator is a detailed report that analyzes simulation results, explains the design rationale, and delves into the technical intricacies of the implementation. This comprehensive documentation provides valuable insights into cache memory behavior, enabling developers and researchers to optimize their cache designs effectively. By sharing our methodology and findings, this project aims to advance understanding and foster collaboration in the fields of cache memory optimization and computer architecture.

DESIGN METHODOLOGY

Understanding cache design and operation is fundamental to mastering efficient memory management techniques. Cache memory is a specialized, high-speed memory that bridges the performance gap between the CPU and slower memory components, such as RAM and disk storage. While less costly than CPU registers, cache memory is more expensive than main memory but offers significantly faster access times. Positioned as a buffer between the CPU and RAM, it stores frequently accessed data and instructions, enabling the CPU to retrieve information quickly and thereby reducing the average time required to access main memory.

Cache memory operates by maintaining copies of frequently accessed data from main memory in a smaller, faster memory space. Modern CPUs typically include separate instruction and data caches to streamline processing. When the CPU requires data, it first checks the cache. If the requested data is found—a cache hit—the data is retrieved directly from the cache. Conversely, a cache miss occurs when the data is absent, prompting the cache to fetch it from main memory, create a new cache entry, and subsequently serve the request.

For instance, consider a system with a main memory comprising 128 words, where each block contains four words. This setup results in 32 memory blocks, each holding four words. If the cache size is limited to 16 words, the cache accommodates four lines (16/4 = 4), with each line corresponding to a block size of four words. The mapping of main memory blocks to cache lines is determined by the modulo operation (k mod n), where k is the block number, and n is the number of cache lines. In a direct-mapped cache, block 4 maps exclusively to line 0, as *4 mod 4 = 0*. Similarly, block 13 maps to line 1 (*13 mod 4 = 1*), irrespective of the occupancy of other lines.

This mapping principle, illustrated in Figure 1, demonstrates how main memory blocks are allocated to cache lines in a direct-mapped configuration. By efficiently managing these mappings and leveraging cache memory, systems achieve enhanced performance through reduced memory access times.

DESCRIPTION OF THE DESIGN

Blocks of different sizes are used to segment caches. The number of blocks in a cache is typically a power of two. The simplest method is the Direct-Mapped Cache. In Direct Mapped Cache, every memory address correlates to a single cache block. One method for figuring out which cache block belongs at a given memory location is the mod (remainder) operation. The data at the memory address would be delivered to the cache block index if the cache contained 2^k blocks, giving I mod 2^k. At any one time, the CPU only needs a specific quantity of data. The cache memory duplicates the contents of the memory locations ahead of time to identify the range of memory locations that the processor will need shortly. The CPU loads data from the main memory into the cache first when it needs it, and if it can't find it there, it must wait until the main memory is filled with the necessary data. At this point, data from nearby sites that match the requested address are also transferred to the cache. The basic idea behind cache operation is the locality of reference, sometimes referred to as the concept of locality. According to this principle, every program or application running on a processor ought to only use a tiny portion of the total address space at any one time. Reference locations come in two varieties.

1. Temporal locality, or time locality: There's a good chance that a certain data item will be needed again shortly if it's utilized at one point in time.

2. Spatial locality (a spatial locality): There is a high probability that data items from nearby addresses will be needed shortly if a data item from a particular memory location is used at a given moment.

A cache controller receives the address when the CPU tries to read from memory. A block in the cache will be indexed using the lowest k bits of the address. If the block is valid and the tag matches the top (m-k) bits of the m-bit address, the data will be sent to the CPU. What we've been expecting to be memory delays are cache hits. If the CPU implementations had gone straight to the main memory, the cycle times would have been much longer. However, a much slower main memory access is necessary upon a cache miss. A total of 2n words and n-bit address lines make up the main memory. It is arranged into several blocks, with k words in each block. Consequently, its total number of main memory blocks is 2n/k = MB. Similarly, a few lines, each containing k words, make up the cache memory. There are far fewer cache lines (CL) than there are main memory blocks (MB) in total. Because of this, only a small number of main memory blocks are mapped to the cache at any given moment. The full block containing the requested data item is mapped to one of the cache lines when the processor starts a read request for data that is not in the cache. A cache line cannot be unique to a single main memory block due to the vast number of blocks compared to cache lines. Consequently, a tag designating the physical location linked to every cache line is assigned. How soon it returns the requested data is what defines the cache's performance. It is computed using the access time and hit/miss ratio. A cache hit occurs when the requested data is found in the cache; a cache miss occurs when it is not. The number of memory accesses that are discovered in the cache relative to the total number of memory accesses requested is known as the hit ratio. One may describe the miss ratio as (1 - hit ratio). Hit time is the amount of time it takes the cache to provide the requested data in the case of a hit. The necessary data is acquired from the main memory and mapped to the cache in the event of a cache miss. The time it

takes to get and map the data is called a miss penalty. Conventional methods of enhancing cache performance (by raising the hit or lowering the miss ratio) are generally categorized as

(1) increasing block and cache size,

(2) increasing associativity,

(3) cache probing,

(4) supplementing the regular cache with victim cache,

(5) hardware data prefetching, and

(6) including additional cache hierarchy.

In the direct mapped cache, or one-way-set associative cache, every block from the main memory is allocated to a particular cache line. There are two benefits to the direct mapped cache: a straightforward design and extremely easy hardware implementation. But this architecture's mapping of every block from main memory to a single cache line is its lone flaw. The Direct Mapped Cache architecture is shown in the following diagram.

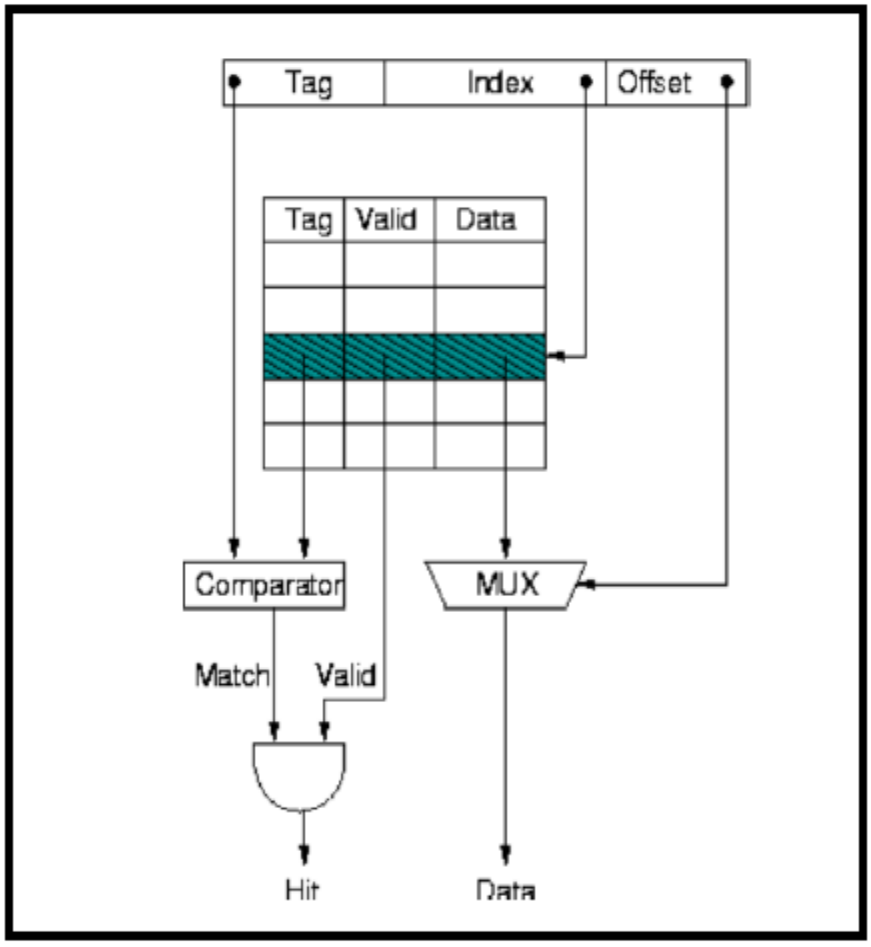


Figure 1: Architecture of Direct Mapped Cache

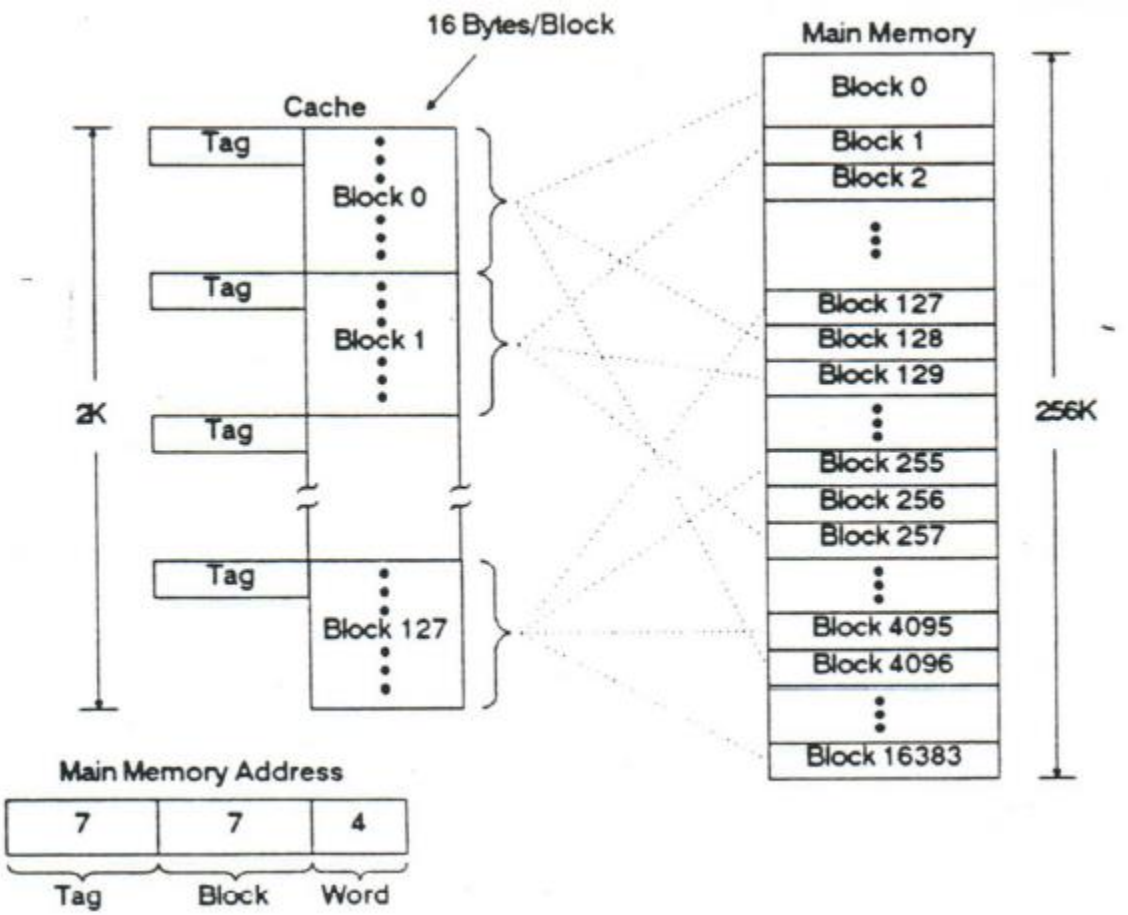


Figure 2: Direct cache Mapping example.

In set-associative cache design, the cache memory is divided into several tiny, direct-mapped modules, each of which is referred to as a set. A cache composed of N sets is referred to as N-way set associative. The performance of the set-associative cache lies in the center of these two cache architectures as it is essentially a compromise between completely associative and direct-mapped caches. Two-way and four-way set-associative caches offer the best performance in terms of hit ratio and access time for embedded systems. A 4-way set associative cache's design is shown in the following diagram:

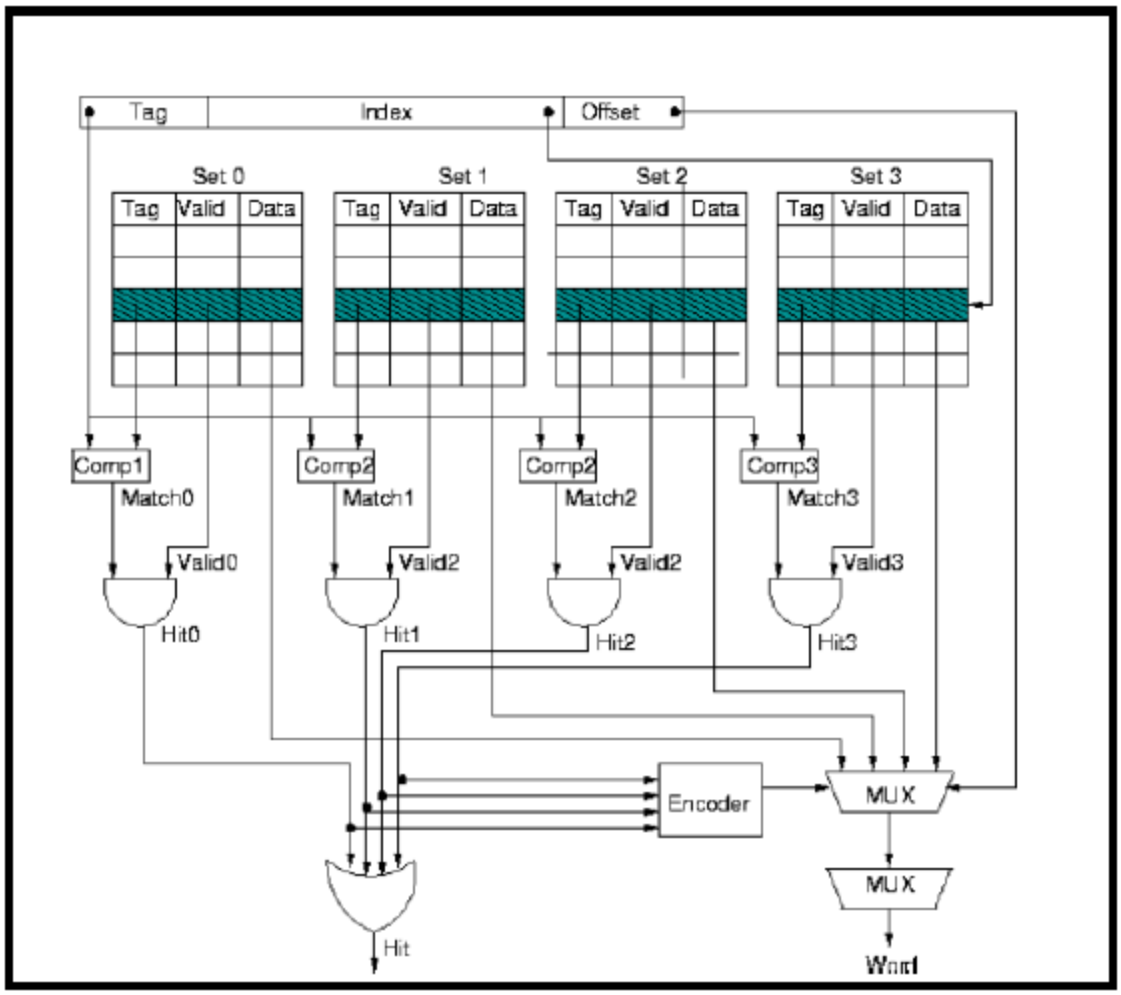


Figure 3: Architecture of 4-way set-associative cache.

In a completely associative cache, every block from the main memory can be allocated to any of the cache lines, giving the best possible hit ratio. It also involves the expense, intricacy, hardware, and access time related to looking up the requested location in the cache. Every cache block is compared simultaneously to determine if a requested memory location is present in the cache. The building of a completely associative cache is shown in the following diagram:

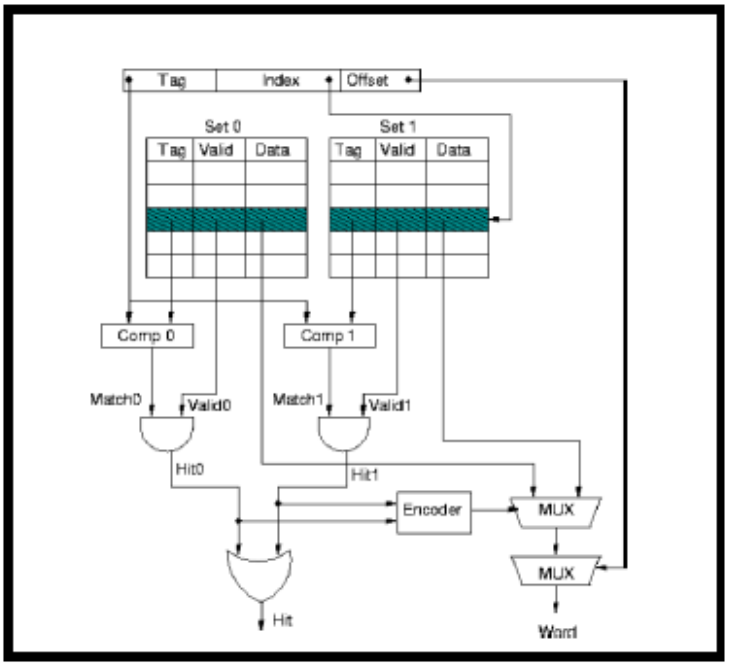


Figure 4: Architecture of fully set-associative cache.

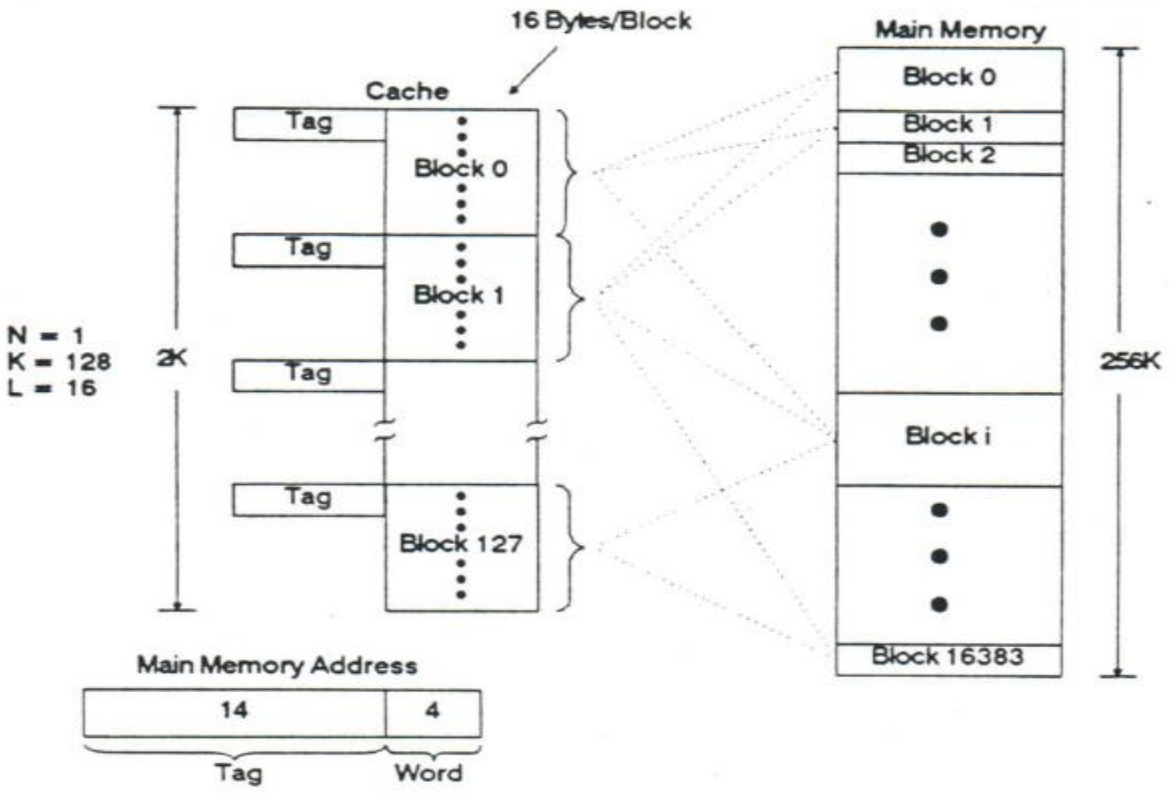


Figure 5: Fully Associative set mapping example.

various types of caching need various hit and access times because of the comparison, as the table below illustrates: In the effort to construct the data cache simulator, data is replaced using an LRU. In the LRU replacement policy, it replaces a cache line that hasn't been utilized in a while.

Direct Mapped	Less	Poor comparatively
N-way Set Associative	Moderate (worsen as N increases)	Good, (improves further as N increases)
Fully Associative	Highest	Best

Table 1: Comparison of Mapping functions

SIMULATION AND VERIFICATION

First, compare the miss ratio to the variable cache size. In the first example, with the fixed block size 32 and fixed associativity 4, a line will be formed with the miss rate as the Y-axis and the cache size as the X-axis for the graph.

Plotting and data collection results for cache sizes of 16KB, 32KB, 64KB, and 128KB are as follows:

S.No.	Cache Size(in KB)	Block size	Set Associativity	Total Access Count	Cache Hits	Hit Ratio	Miss Ratio
1	**16**	32	4	1500000	1486499	99.1	**0.9**
2	**32**	32	4	1500000	1492446	99.5	**0.5**
3	**64**	32	4	1500000	1494746	99.65	**0.35**
4	**128**	32	4	1500000	1494143	99.79	**0.21**

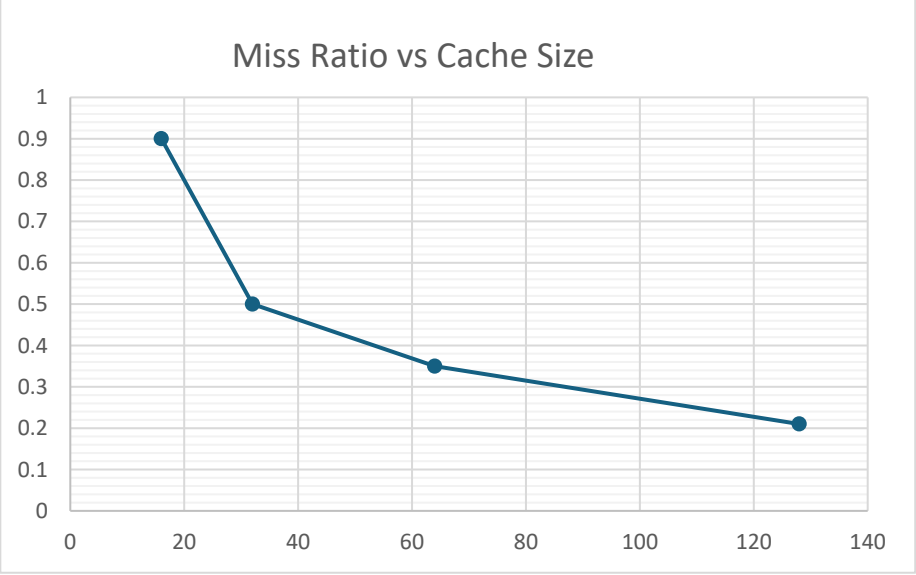


Figure 6: Miss Ratio Vs Cache Size Chart

Furthermore, it is important to consider the changeable line size of the Miss Ratio. In the second example, given the fixed cache size of 8192 and fixed associativity of 4, a line will be formed with

the block size as the X-axis and the miss rate as the Y-axis for the graph. The results are obtained, and a graph based on the results for block sizes 16, 32, 64, and 128 is produced, as shown in the diagram below.

S.No.	Cache Size(in KB)	Block size	Set Associativity	Total Access Count	Cache Hits	Hit Ratio	Miss Ratio
1	8192	**16**	4	1500000	1464178	97.79	**2.21**
2	8192	**32**	4	1500000	1459535	97.3	**2.7**
3	8192	**64**	4	1500000	1436714	95.78	**4.22**
4	8192	**128**	4	1500000	1387034	92.47	**7.53**

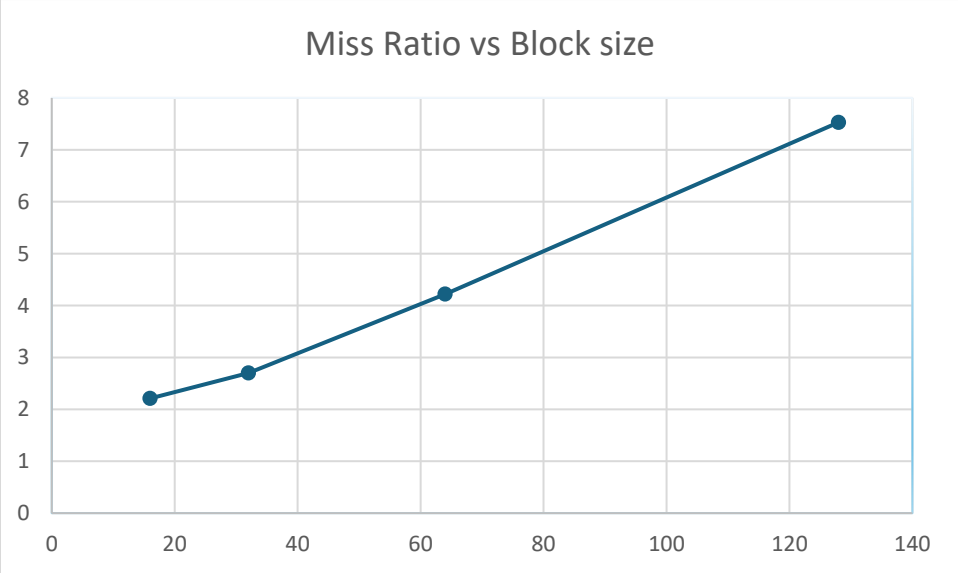


Figure 7: Miss Ratio Vs Block Size Chart

Third, contrast the Miss Ratio with Variable Associativity. In the third example, given the fixed cache size of 8192 and the fixed block size of 32, a line will be drawn with the associativity acting as the X-axis and the miss rate acting as the Y-axis for the graphs. The results for associativity of 1, 2, 4, 8, and completely associative are generated and displayed, as seen in the figure below.

S.No.	Cache Size(in KB)	Block size	Set Associativity	Total Access Count	Cache Hits	Hit Ratio	Miss Ratio
1	8192	32	**1**	1500000	1365252	91.02	**8.98**
2	8192	32	**2**	1500000	1433656	95.58	**4.42**
3	8192	32	**4**	1500000	1459535	97.3	**2.7**
4	8192	32	**8**	1500000	1470136	98.01	**1.99**
5	8192	32	**32**	1500000	1475433	98.36	**1.64**

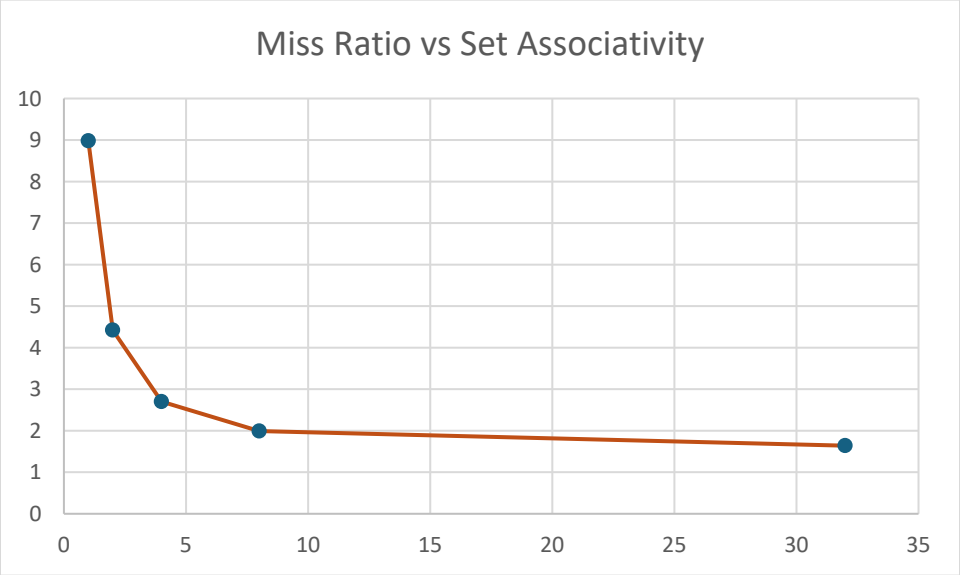


Figure 8: Miss Ratio Vs Associativity Chart

CONCLUSION

The research focused on developing a data cache simulator capable of handling m-way set-associative caches with configurations of m=1,2,4,8,16,32, and fully associative caches. Using Verilog and the Least Recently Used (LRU) algorithm, the simulator was successfully implemented, requiring 2GB of RAM for the given address space. The study explored three primary cache parameters—associativity, line size, and cache size—and analyzed their influence on the system's hit and miss rates.

Cache size emerged as a vital determinant of cache performance. Increasing the cache size generally led to a higher hit rate by reducing capacity misses, as more blocks could be stored in the cache. However, larger cache sizes also introduced higher storage costs, underscoring the need to balance cache capacity and expense to achieve efficient performance. The study highlighted that determining the optimal cache size requires careful consideration of both performance and cost constraints.

Line size, which defines the amount of data transferred from main memory to the cache during a miss, also significantly impacted cache performance. Larger line sizes tended to improve the hit ratio by exploiting spatial locality, as fetching adjacent data reduced the need for additional memory accesses. However, when the line size became excessively large, the hit ratio declined because newly fetched blocks might contain unused data, which displaced frequently accessed blocks. This result emphasizes the importance of selecting an appropriate line size to balance spatial locality and block utilization.

Associativity was another critical factor examined in the research. Higher associativity allowed memory blocks to be mapped to multiple cache lines within a set, which reduced conflict misses and improved the overall miss rate. Fully associative caches demonstrated the lowest miss rates due to their flexibility in block placement. However, this configuration is more costly and complex compared to lower associativities, highlighting the trade-offs involved in cache design.

The experimental results revealed how cache size, line size, and associativity interact to influence hit and miss rates. Larger cache sizes and higher associativity typically improved performance by reducing misses, while appropriately sized lines enhanced the hit ratio by leveraging spatial locality. These findings provide valuable insights into designing efficient and cost-effective cache systems while balancing performance with implementation complexity.

APPENDIX

DESIGN FILE

```
`define cache_size (1024*8)
`define line_size 128
`define Associativity 4

`define Index_bit (`Associativity==0)? 0: $clog2(`cache_size/(`line_size*`Associativity))
`define Offset_bit $clog2(`line_size)
`define Tag_bit 31-(`Offset_bit+`Index_bit)



module test(adder_41,clk_41, rst_41,misses_41,hits_41);
  input clk_41, rst_41;
  input [30:0] adder_41;
  output [30:0] misses_41,hits_41;

  reg [30:0] Num_Blocks_41,Num_Sets_41,misses_41,hits_41,cache_block_41,cache_set_41,set_index_41, Curr_Block_41,Curr_Count_41;
  reg data_present_41;

  integer i,j;
  integer k;

  parameter CS = `cache_size;
  parameter LS = `line_size;
  parameter Assoc = (`Associativity==0)? (CS/LS):`Associativity;
  parameter Tbit = `Tag_bit;
  parameter Ob = `Offset_bit;
  parameter Ib = `Index_bit;
```

```
reg [(LS*8)-1:0] cache [0:(CS/LS) - 1];
reg [Tbit-1:0] tag_array [0:(CS/LS) - 1]; // for all tags in cache
reg valid_array [0:(CS/LS) - 1]; //0 - there is no data 1 - there is data


reg [Tbit-1:0] tag; // for current tag

reg [Assoc-1:0] counter [0:(CS/LS) - 1];


initial begin
 hits_41 = 0;
 misses_41 = 0;
 Num_Blocks_41 = CS/LS;
 Num_Sets_41 = Num_Blocks_41/Assoc;

 for (k = 0; k <  Num_Blocks_41 ; k = k + 1)
                begin
       valid_array[k] = 0;
       tag_array[k] = 0;
                end
 for (j = 0; j <  Num_Sets_41 ; j = j + 1)
   for (k = 0; k <  Assoc ; k = k + 1)
     counter[(j*Assoc)+k] = k;

end
```

```
always@(posedge clk_41)begin

 cache_block_41 = (adder_41/LS)% Num_Blocks_41;
 cache_set_41 = ((adder_41/LS)% (Num_Blocks_41/Assoc));
 set_index_41 = cache_set_41*Assoc; //pointing at first_41 block of current set
 data_present_41 = 0;

 tag = adder_41[30:(Ob+Ib)];

 for(i=0; i<Assoc; i=i+1) begin
  if ((valid_array [set_index_41+i] == 1) && (tag == tag_array[set_index_41+i])) begin
  hits_41 = hits_41+1;
  data_present_41 = 1;
  Curr_Block_41 = set_index_41+i;
  Curr_Count_41 = counter[Curr_Block_41];
 // $display("hits_41=%d",hits_41);
  end
 end

 if (data_present_41 == 0) begin
  misses_41 = misses_41+1;
 //  $display("misses_41=%d",misses_41);

  for(i=0; i<Assoc; i=i+1) begin
    if (counter[set_index_41+i]==0)begin
                tag_array[set_index_41+i] = tag;
                valid_array [set_index_41+i] = 1;
                Curr_Block_41 = set_index_41+i;
                Curr_Count_41 = 0;
    end
```

```
    end
  end

    for(i=0; i<Assoc; i=i+1) begin // counter

      if (counter[set_index_41+i]>Curr_Count_41)begin
        counter[set_index_41+i] = counter[set_index_41+i] - 1;
      end

      counter[Curr_Block_41] = Assoc - 1 ;

    end

 end



endmodule
```

TESTBENCH FILE

```
module tb();
 reg clk_41, rst_41;
 reg [30:0] adder_41;
 reg [8:0]LS_41;
 reg [16:0] CS_41;
 reg [5:0] Assoc_41;

  wire [30:0] misses_41,hits_41;
 real hit_ratio_41,per = 100.00;
 real m,h;

 `define Length 1500000
        integer p,r,c;
  integer file,stat,out,i,j;
 reg signed [30:0] face[0:`Length-1];
 reg [30:0] actual[0:`Length-1];

 test t1(adder_41,clk_41, rst_41,misses_41,hits_41);

 initial begin
  file=$fopen("addr_trace.txt","r");
            i=0;
        while (! $feof(file))
           begin

              stat=$fscanf(file,"%d\n",face[i]);
                    i=i+1;
           end
```

```
        $fclose(file);



  for(i=0;i<`Length;i=i+1)
            begin
              if (i==0)  actual[0] = face[0];
              else actual[i]=actual[i-1]+face[i];

            end
end

always  #5 clk_41 = ~clk_41;

initial begin

  clk_41 = 0;


  for (j = 0 ; j<`Length ; j=j+1) begin


    @(posedge clk_41) adder_41 = actual[j];
    //$display("Address=%d",adder_41);
  end


  @(posedge clk_41)
  m=misses_41;
```

```
    h=hits_41;
   hit_ratio_41 = (h/(m+h))*per;
    $display("Hit Ratio=%7.2f Percentage, Cache Hits = %d, Total Simulation Addresses =
%d",hit_ratio_41,hits_41,(misses_41 + hits_41));
    $finish;
  end



endmodule
```

SIMULATION RESULTS :

First case: Varying Cache size (Fixed block size= 32 Bytes, Fixed Associativity=4)
Cache size = 16KB:

```
File  Edit  View  Search  Terminal  Help
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
        However, due to incremental compilation, only 1 module needs to be compiled.
recompiling module tb because:
        Module parameters have been changed via defparam.
rm -f _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_l.o
ld -m elf_i386 -shared -o .//../simv.daidir//pre_vcsobj_1_1.so --whole-archive pre_vcsobj_1_1.a --no-whole-archive
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32 -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-rpath='$ORIGIN'/simv.daidir/
/scsim.db.dir  _csrc1.so _csrc0.so   pre_vcsobj_1_1.so  rmapats_mop.o rmapats.o rmar.o           /apps/synopsys/VCS
MX_NEW/linux/lib/libzerosoft_rt_stubs.so /apps/synopsys/VCSMX_NEW/linux/lib/libvirsim.so /apps/synopsys/VCSMX_NEW/l
inux/lib/librterrorinf.so /apps/synopsys/VCSMX_NEW/linux/lib/libsnpsmalloc.so    /apps/synopsys/VCSMX_NEW/linux/lib
/libvcsnew.so /apps/synopsys/VCSMX_NEW/linux/lib/libuclinative.so   -Wl,-whole-archive /apps/synopsys/VCSMX_NEW/lin
ux/lib/libvcsucli.so -Wl,-no-whole-archive          /apps/synopsys/VCSMX_NEW/linux/lib/vcs_save_restore_new.o /apps
/synopsys/VCSMX_NEW/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .095 seconds to compile + .018 seconds to elab + .106 seconds to link
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 23:01 2024
Hit Ratio=  99.10 Percentage, Cache Hits =    1486499, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time             15000005
           V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.210 seconds;       Data structure size:  22.9Mb
Wed Apr 17 23:01:19 2024
[017410341@coe-ee-cad50 mini3]$
```

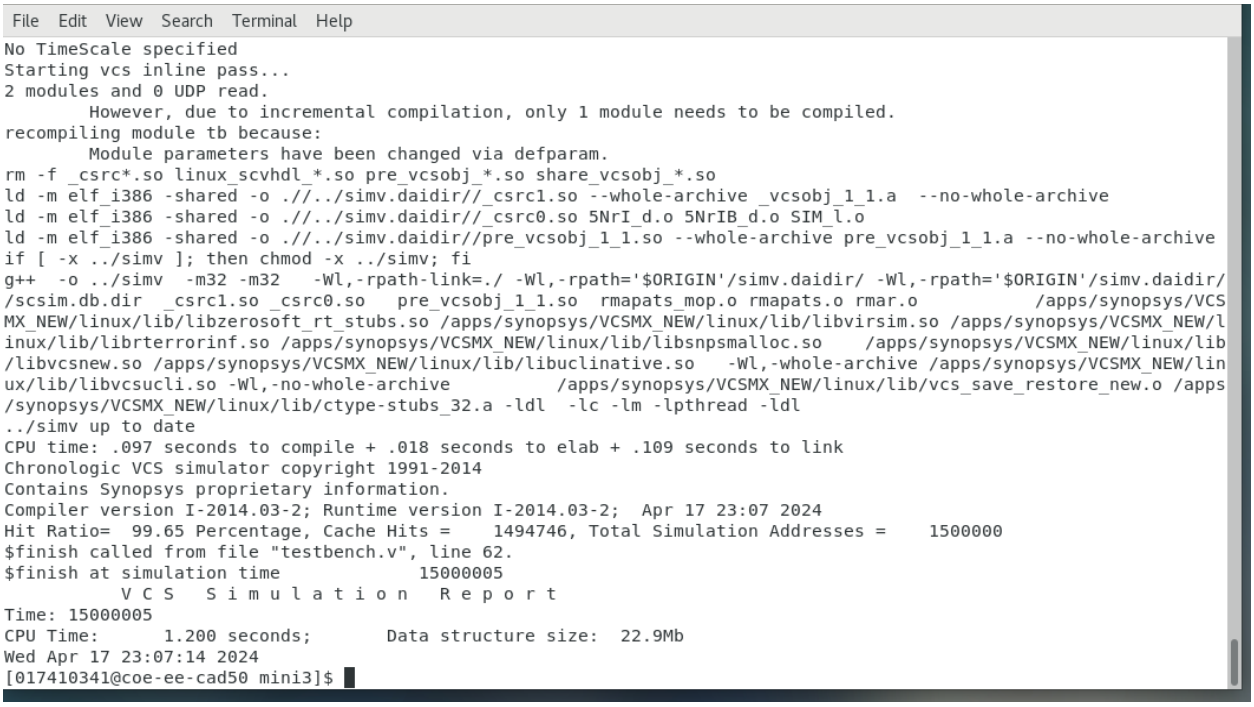
Varying Cache size (Fixed block size= 32 Bytes, Fixed Associativity=4)

Cache size = 32KB:

```
File  Edit  View  Search  Terminal  Help
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
        However, due to incremental compilation, only 1 module needs to be compiled.
recompiling module tb because:
        Module parameters have been changed via defparam.
rm -f _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_l.o
ld -m elf_i386 -shared -o .//../simv.daidir//pre_vcsobj_1_1.so --whole-archive pre_vcsobj_1_1.a --no-whole-archive
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32 -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-rpath='$ORIGIN'/simv.daidir/
/scsim.db.dir  _csrc1.so _csrc0.so   pre_vcsobj_1_1.so  rmapats_mop.o rmapats.o rmar.o           /apps/synopsys/VCS
MX_NEW/linux/lib/libzerosoft_rt_stubs.so /apps/synopsys/VCSMX_NEW/linux/lib/libvirsim.so /apps/synopsys/VCSMX_NEW/l
inux/lib/librterrorinf.so /apps/synopsys/VCSMX_NEW/linux/lib/libsnpsmalloc.so    /apps/synopsys/VCSMX_NEW/linux/lib
/libvcsnew.so /apps/synopsys/VCSMX_NEW/linux/lib/libuclinative.so   -Wl,-whole-archive /apps/synopsys/VCSMX_NEW/lin
ux/lib/libvcsucli.so -Wl,-no-whole-archive          /apps/synopsys/VCSMX_NEW/linux/lib/vcs_save_restore_new.o /apps
/synopsys/VCSMX_NEW/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .101 seconds to compile + .018 seconds to elab + .099 seconds to link
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 23:04 2024
Hit Ratio=  99.50 Percentage, Cache Hits =    1492446, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time             15000005
           V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.180 seconds;       Data structure size:  22.9Mb
Wed Apr 17 23:04:39 2024
[017410341@coe-ee-cad50 mini3]$
```

Varying Cache size (Fixed block size= 32 Bytes, Fixed Associativity=4)

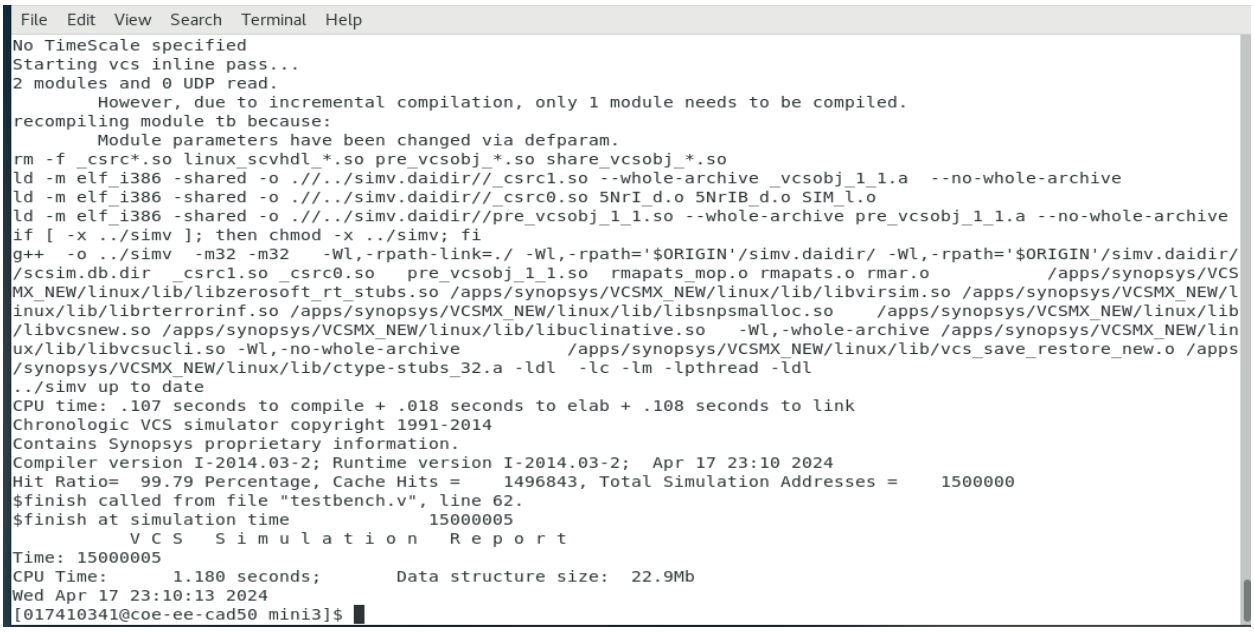
Cache size = 64KB:



```
File  Edit  View  Search  Terminal  Help
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
        However, due to incremental compilation, only 1 module needs to be compiled.
recompiling module tb because:
        Module parameters have been changed via defparam.
rm -f _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_l.o
ld -m elf_i386 -shared -o .//../simv.daidir//pre_vcsobj_1_1.so --whole-archive pre_vcsobj_1_1.a --no-whole-archive
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32 -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-rpath='$ORIGIN'/simv.daidir/
/scsim.db.dir  _csrc1.so _csrc0.so   pre_vcsobj_1_1.so  rmapats_mop.o rmapats.o rmar.o           /apps/synopsys/VCS
MX_NEW/linux/lib/libzerosoft_rt_stubs.so /apps/synopsys/VCSMX_NEW/linux/lib/libvirsim.so /apps/synopsys/VCSMX_NEW/l
inux/lib/librterrorinf.so /apps/synopsys/VCSMX_NEW/linux/lib/libsnpsmalloc.so    /apps/synopsys/VCSMX_NEW/linux/lib
/libvcsnew.so /apps/synopsys/VCSMX_NEW/linux/lib/libuclinative.so   -Wl,-whole-archive /apps/synopsys/VCSMX_NEW/lin
ux/lib/libvcsucli.so -Wl,-no-whole-archive          /apps/synopsys/VCSMX_NEW/linux/lib/vcs_save_restore_new.o /apps
/synopsys/VCSMX_NEW/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .097 seconds to compile + .018 seconds to elab + .109 seconds to link
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 23:07 2024
Hit Ratio=  99.65 Percentage, Cache Hits =    1494746, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time             15000005
          V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.200 seconds;       Data structure size:  22.9Mb
Wed Apr 17 23:07:14 2024
[017410341@coe-ee-cad50 mini3]$
```

Varying Cache size (Fixed block size= 32 Bytes, Fixed Associativity=4)

Cache size = 128KB:



```
File  Edit  View  Search  Terminal  Help
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
        However, due to incremental compilation, only 1 module needs to be compiled.
recompiling module tb because:
        Module parameters have been changed via defparam.
rm -f _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_l.o
ld -m elf_i386 -shared -o .//../simv.daidir//pre_vcsobj_1_1.so --whole-archive pre_vcsobj_1_1.a --no-whole-archive
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32 -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-rpath='$ORIGIN'/simv.daidir/
/scsim.db.dir  _csrc1.so _csrc0.so   pre_vcsobj_1_1.so  rmapats_mop.o rmapats.o rmar.o           /apps/synopsys/VCS
MX_NEW/linux/lib/libzerosoft_rt_stubs.so /apps/synopsys/VCSMX_NEW/linux/lib/libvirsim.so /apps/synopsys/VCSMX_NEW/l
inux/lib/librterrorinf.so /apps/synopsys/VCSMX_NEW/linux/lib/libsnpsmalloc.so    /apps/synopsys/VCSMX_NEW/linux/lib
/libvcsnew.so /apps/synopsys/VCSMX_NEW/linux/lib/libuclinative.so   -Wl,-whole-archive /apps/synopsys/VCSMX_NEW/lin
ux/lib/libvcsucli.so -Wl,-no-whole-archive          /apps/synopsys/VCSMX_NEW/linux/lib/vcs_save_restore_new.o /apps
/synopsys/VCSMX_NEW/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .107 seconds to compile + .018 seconds to elab + .108 seconds to link
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 23:10 2024
Hit Ratio=  99.79 Percentage, Cache Hits =    1496843, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time             15000005
          V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.180 seconds;       Data structure size:  22.9Mb
Wed Apr 17 23:10:13 2024
[017410341@coe-ee-cad50 mini3]$
```

Second case: Varying Block size (fixed cache size = 8192 KB, fixed associativity =4)

block size = 16:

```
File  Edit  View  Search  Terminal  Help
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
        However, due to incremental compilation, only 1 module needs to be compiled.
recompiling module tb because:
        Module parameters have been changed via defparam.
rm -f _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_l.o
ld -m elf_i386 -shared -o .//../simv.daidir//pre_vcsobj_1_1.so --whole-archive pre_vcsobj_1_1.a --no-whole-archive
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32 -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-rpath='$ORIGIN'/simv.daidir/
/scsim.db.dir  _csrc1.so _csrc0.so   pre_vcsobj_1_1.so  rmapats_mop.o rmapats.o rmar.o           /apps/synopsys/VCS
MX_NEW/linux/lib/libzerosoft_rt_stubs.so /apps/synopsys/VCSMX_NEW/linux/lib/libvirsim.so /apps/synopsys/VCSMX_NEW/l
inux/lib/librterrorinf.so /apps/synopsys/VCSMX_NEW/linux/lib/libsnpsmalloc.so    /apps/synopsys/VCSMX_NEW/linux/lib
/libvcsnew.so /apps/synopsys/VCSMX_NEW/linux/lib/libuclinative.so   -Wl,-whole-archive /apps/synopsys/VCSMX_NEW/lin
ux/lib/libvcsucli.so -Wl,-no-whole-archive          /apps/synopsys/VCSMX_NEW/linux/lib/vcs_save_restore_new.o /apps
/synopsys/VCSMX_NEW/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .098 seconds to compile + .017 seconds to elab + .112 seconds to link
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 22:16 2024
Hit Ratio=  97.79 Percentage, Cache Hits =    1466878, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time              15000005
          V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.200 seconds;       Data structure size:  22.9Mb
Wed Apr 17 22:16:58 2024
[017410341@coe-ee-cad50 mini3]$
```

Block size = 32:

```
File  Edit  View  Search  Terminal  Help
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
        However, due to incremental compilation, only 1 module needs to be compiled.
recompiling module tb because:
        Module parameters have been changed via defparam.
rm -f _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_l.o
ld -m elf_i386 -shared -o .//../simv.daidir//pre_vcsobj_0_1.so --whole-archive pre_vcsobj_0_1.a --no-whole-archive
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32 -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-rpath='$ORIGIN'/simv.daidir/
/scsim.db.dir  _csrc1.so _csrc0.so   pre_vcsobj_0_1.so  rmapats_mop.o rmapats.o rmar.o           /apps/synopsys/VCS
MX_NEW/linux/lib/libzerosoft_rt_stubs.so /apps/synopsys/VCSMX_NEW/linux/lib/libvirsim.so /apps/synopsys/VCSMX_NEW/l
inux/lib/librterrorinf.so /apps/synopsys/VCSMX_NEW/linux/lib/libsnpsmalloc.so    /apps/synopsys/VCSMX_NEW/linux/lib
/libvcsnew.so /apps/synopsys/VCSMX_NEW/linux/lib/libuclinative.so   -Wl,-whole-archive /apps/synopsys/VCSMX_NEW/lin
ux/lib/libvcsucli.so -Wl,-no-whole-archive          /apps/synopsys/VCSMX_NEW/linux/lib/vcs_save_restore_new.o /apps
/synopsys/VCSMX_NEW/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .099 seconds to compile + .018 seconds to elab + .114 seconds to link
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 22:14 2024
Hit Ratio=  97.30 Percentage, Cache Hits =    1459535, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time             15000005
          V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.290 seconds;       Data structure size:  22.9Mb
Wed Apr 17 22:14:08 2024
[017410341@coe-ee-cad50 mini3]$
```

Block size = 64:

```
File  Edit  View  Search  Terminal  Help
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
        However, due to incremental compilation, only 1 module needs to be compiled.
recompiling module tb because:
        Module parameters have been changed via defparam.
rm -f _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_l.o
ld -m elf_i386 -shared -o .//../simv.daidir//pre_vcsobj_1_1.so --whole-archive pre_vcsobj_1_1.a --no-whole-archive
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32 -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-rpath='$ORIGIN'/simv.daidir/
/scsim.db.dir  _csrc1.so _csrc0.so   pre_vcsobj_1_1.so  rmapats_mop.o rmapats.o rmar.o           /apps/synopsys/VCS
MX_NEW/linux/lib/libzerosoft_rt_stubs.so /apps/synopsys/VCSMX_NEW/linux/lib/libvirsim.so /apps/synopsys/VCSMX_NEW/l
inux/lib/librterrorinf.so /apps/synopsys/VCSMX_NEW/linux/lib/libsnpsmalloc.so    /apps/synopsys/VCSMX_NEW/linux/lib
/libvcsnew.so /apps/synopsys/VCSMX_NEW/linux/lib/libuclinative.so   -Wl,-whole-archive /apps/synopsys/VCSMX_NEW/lin
ux/lib/libvcsucli.so -Wl,-no-whole-archive          /apps/synopsys/VCSMX_NEW/linux/lib/vcs_save_restore_new.o /apps
/synopsys/VCSMX_NEW/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .098 seconds to compile + .017 seconds to elab + .112 seconds to link
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 22:16 2024
Hit Ratio=  97.79 Percentage, Cache Hits =    1466878, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time             15000005
          V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.200 seconds;       Data structure size:  22.9Mb
Wed Apr 17 22:16:58 2024
[017410341@coe-ee-cad50 mini3]$
```

Block size = 128:

```
File  Edit  View  Search  Terminal  Help
  Please refer to release notes for information on supported platforms.


Warning-[LINX_KRNL] Unsupported Linux kernel
  Linux kernel '3.10.0-1160.108.1.el7.x86_64' is not supported.
  Supported versions are 2.4* or 2.6*.

                        Chronologic VCS (TM)
          Version I-2014.03-2 -- Wed Apr 17 22:07:04 2024
              Copyright (c) 1991-2014 by Synopsys Inc.
                        ALL RIGHTS RESERVED

This program is proprietary and confidential information of Synopsys Inc.
and may be used and disclosed only as authorized in a license agreement
controlling such use and disclosure.

The design hasn't changed and need not be recompiled.
If you really want to, delete file simv.daidir/.vcs.timestamp and
run VCS again.

Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 22:07 2024
Hit Ratio=  92.47 Percentage, Cache Hits =    1387034, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time             15000005
          V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.190 seconds;       Data structure size:  22.9Mb
Wed Apr 17 22:07:06 2024
[017410341@coe-ee-cad50 mini3]$
```

Third case: Varying Associativity (fixed cache size = 8192 KB, block size =32KB)

1-way Set Associativity (Direct mapped cache):

```
File  Edit  View  Search  Terminal  Help
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
        However, due to incremental compilation, only 1 module needs to be compiled.
recompiling module tb because:
        Module parameters have been changed via defparam.
rm -f _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_l.o
ld -m elf_i386 -shared -o .//../simv.daidir//pre_vcsobj_1_1.so --whole-archive pre_vcsobj_1_1.a --no-whole-archive
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32 -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-rpath='$ORIGIN'/simv.daidir/
/scsim.db.dir  _csrc1.so _csrc0.so   pre_vcsobj_1_1.so  rmapats_mop.o rmapats.o rmar.o           /apps/synopsys/VCS
MX_NEW/linux/lib/libzerosoft_rt_stubs.so /apps/synopsys/VCSMX_NEW/linux/lib/libvirsim.so /apps/synopsys/VCSMX_NEW/l
inux/lib/librterrorinf.so /apps/synopsys/VCSMX_NEW/linux/lib/libsnpsmalloc.so    /apps/synopsys/VCSMX_NEW/linux/lib
/libvcsnew.so /apps/synopsys/VCSMX_NEW/linux/lib/libuclinative.so   -Wl,-whole-archive /apps/synopsys/VCSMX_NEW/lin
ux/lib/libvcsucli.so -Wl,-no-whole-archive          /apps/synopsys/VCSMX_NEW/linux/lib/vcs_save_restore_new.o /apps
/synopsys/VCSMX_NEW/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .097 seconds to compile + .018 seconds to elab + .110 seconds to link
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 23:17 2024
Hit Ratio=  91.02 Percentage, Cache Hits =    1365252, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time             15000005
           V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.060 seconds;       Data structure size:  22.9Mb
Wed Apr 17 23:17:51 2024
[017410341@coe-ee-cad50 mini3]$
```

2-way set associativity:

```
File  Edit  View  Search  Terminal  Help
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
        However, due to incremental compilation, only 1 module needs to be compiled.
recompiling module tb because:
        Module parameters have been changed via defparam.
rm -f _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_l.o
ld -m elf_i386 -shared -o .//../simv.daidir//pre_vcsobj_1_1.so --whole-archive pre_vcsobj_1_1.a --no-whole-archive
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32 -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-rpath='$ORIGIN'/simv.daidir/
/scsim.db.dir  _csrc1.so _csrc0.so   pre_vcsobj_1_1.so  rmapats_mop.o rmapats.o rmar.o           /apps/synopsys/VCS
MX_NEW/linux/lib/libzerosoft_rt_stubs.so /apps/synopsys/VCSMX_NEW/linux/lib/libvirsim.so /apps/synopsys/VCSMX_NEW/l
inux/lib/librterrorinf.so /apps/synopsys/VCSMX_NEW/linux/lib/libsnpsmalloc.so    /apps/synopsys/VCSMX_NEW/linux/lib
/libvcsnew.so /apps/synopsys/VCSMX_NEW/linux/lib/libuclinative.so   -Wl,-whole-archive /apps/synopsys/VCSMX_NEW/lin
ux/lib/libvcsucli.so -Wl,-no-whole-archive          /apps/synopsys/VCSMX_NEW/linux/lib/vcs_save_restore_new.o /apps
/synopsys/VCSMX_NEW/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .108 seconds to compile + .019 seconds to elab + .114 seconds to link
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 23:20 2024
Hit Ratio=  95.58 Percentage, Cache Hits =    1433656, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time             15000005
           V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.310 seconds;       Data structure size:  22.9Mb
Wed Apr 17 23:20:42 2024
[017410341@coe-ee-cad50 mini3]$
```

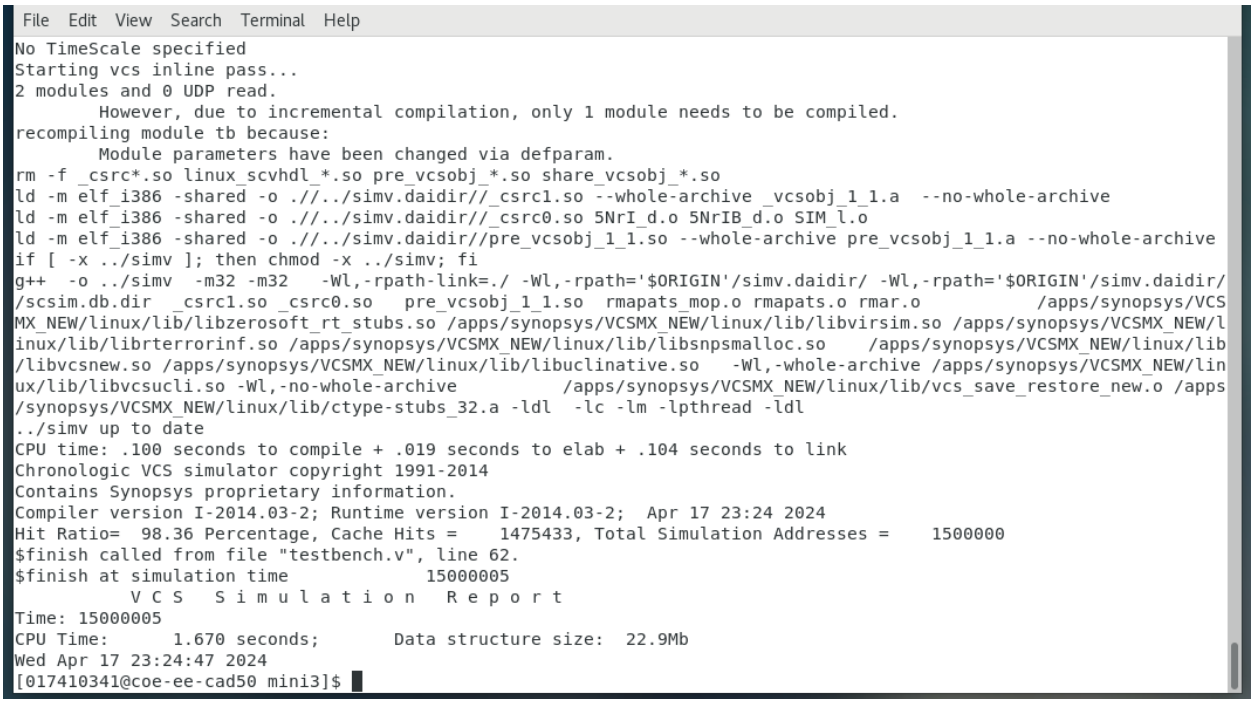
4-way set associativity:

```
File  Edit  View  Search  Terminal  Help
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
        However, due to incremental compilation, only 1 module needs to be compiled.
recompiling module tb because:
        Module parameters have been changed via defparam.
rm -f _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_l.o
ld -m elf_i386 -shared -o .//../simv.daidir//pre_vcsobj_1_1.so --whole-archive pre_vcsobj_1_1.a --no-whole-archive
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32 -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-rpath='$ORIGIN'/simv.daidir/
/scsim.db.dir  _csrc1.so _csrc0.so   pre_vcsobj_1_1.so  rmapats_mop.o rmapats.o rmar.o           /apps/synopsys/VCS
MX_NEW/linux/lib/libzerosoft_rt_stubs.so /apps/synopsys/VCSMX_NEW/linux/lib/libvirsim.so /apps/synopsys/VCSMX_NEW/l
inux/lib/librterrorinf.so /apps/synopsys/VCSMX_NEW/linux/lib/libsnpsmalloc.so    /apps/synopsys/VCSMX_NEW/linux/lib
/libvcsnew.so /apps/synopsys/VCSMX_NEW/linux/lib/libuclinative.so   -Wl,-whole-archive /apps/synopsys/VCSMX_NEW/lin
ux/lib/libvcsucli.so -Wl,-no-whole-archive          /apps/synopsys/VCSMX_NEW/linux/lib/vcs_save_restore_new.o /apps
/synopsys/VCSMX_NEW/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .092 seconds to compile + .019 seconds to elab + .109 seconds to link
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 23:22 2024
Hit Ratio=  97.30 Percentage, Cache Hits =    1459535, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time             15000005
           V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.290 seconds;       Data structure size:  22.9Mb
Wed Apr 17 23:22:20 2024
[017410341@coe-ee-cad50 mini3]$
```

8-way set associativity:

```
File  Edit  View  Search  Terminal  Help
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
        However, due to incremental compilation, only 1 module needs to be compiled.
recompiling module tb because:
        Module parameters have been changed via defparam.
rm -f _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_l.o
ld -m elf_i386 -shared -o .//../simv.daidir//pre_vcsobj_1_1.so --whole-archive pre_vcsobj_1_1.a --no-whole-archive
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32 -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-rpath='$ORIGIN'/simv.daidir/
/scsim.db.dir  _csrc1.so _csrc0.so   pre_vcsobj_1_1.so  rmapats_mop.o rmapats.o rmar.o           /apps/synopsys/VCS
MX_NEW/linux/lib/libzerosoft_rt_stubs.so /apps/synopsys/VCSMX_NEW/linux/lib/libvirsim.so /apps/synopsys/VCSMX_NEW/l
inux/lib/librterrorinf.so /apps/synopsys/VCSMX_NEW/linux/lib/libsnpsmalloc.so    /apps/synopsys/VCSMX_NEW/linux/lib
/libvcsnew.so /apps/synopsys/VCSMX_NEW/linux/lib/libuclinative.so   -Wl,-whole-archive /apps/synopsys/VCSMX_NEW/lin
ux/lib/libvcsucli.so -Wl,-no-whole-archive          /apps/synopsys/VCSMX_NEW/linux/lib/vcs_save_restore_new.o /apps
/synopsys/VCSMX_NEW/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .097 seconds to compile + .019 seconds to elab + .110 seconds to link
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Apr 17 23:23 2024
Hit Ratio=  98.01 Percentage, Cache Hits =    1470136, Total Simulation Addresses =    1500000
$finish called from file "testbench.v", line 62.
$finish at simulation time             15000005
           V C S   S i m u l a t i o n   R e p o r t
Time: 15000005
CPU Time:      1.450 seconds;       Data structure size:  22.9Mb
Wed Apr 17 23:23:34 2024
[017410341@coe-ee-cad50 mini3]$
```

32-way set associativity (Fully Associative):



REFERENCES:

1. Lecture Notes: Dr. Harish Hiriyannaiah
2. Computer Organization and Design: The Hardware/Software Interface,
   Fourth Edition by John L. Hennessy & David A. Patterson
3. Computer system architecture textbook by M Morris Mano ISBN: 81-317-0070-4
4. Cache Memory in Computer Organization - GeeksforGeeks