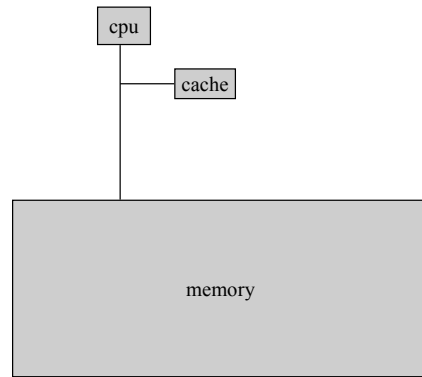


Memory Cache



CSE 240A

Dean Tullsen

Memory Locality

- Memory hierarchies take advantage of *memory locality*.
- *Memory locality* is the principle that future memory accesses are *near* past accesses.
- Memory hierarchies take advantage of two types of locality
 - **Temporal locality** -- near in time => we will often access the same data again very soon
 - **Spatial locality** -- near in space/distance => our next access is often very close to our last access (or recent accesses).

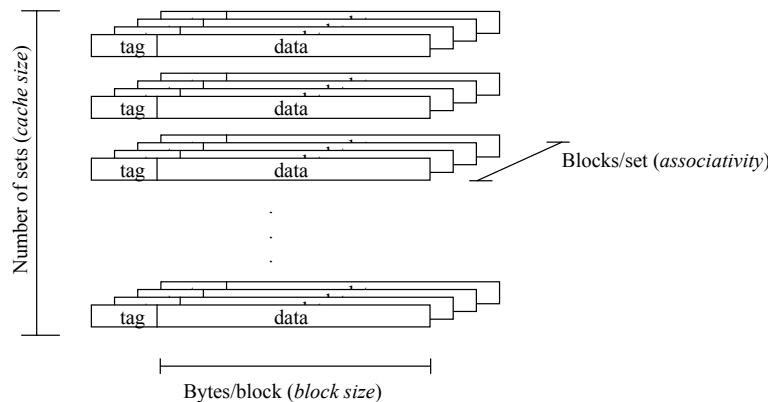
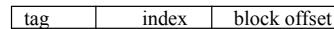
1,2,3,1,2,3,8,8,47,9,10,8,8...

CSE 240A

Dean Tullsen

Cache Organization -- Overview

- A typical cache has three dimensions

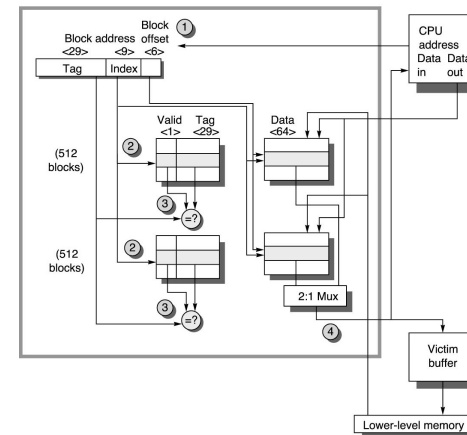


CSE 240A

Dean Tullsen

21264 L1 Data Cache

- 64 KB, 64-byte blocks, 2-way set associative, ? blocks, ? sets
- write-back



CSE 240A

Dean Tullsen

Improving Cache Performance

Average memory-access time = Hit time + Miss rate x Miss penalty (ns or clocks)

- 1. Reduce the miss rate,
- 2. Reduce the miss penalty, or
- 3. Reduce the time to hit in the cache.

Reducing Misses

- Classifying Misses: 3 Cs
 - *Compulsory*—The first access to a block is not in the cache, so the block must be brought into the cache. These are also called *cold start misses* or *first reference misses*.
 - *Capacity*—If C is the size of the cache (in blocks) and there have been more than C unique cache blocks accessed since this cache was last accessed.
 - *Conflict*—Any miss that is not a compulsory miss or capacity miss must be a byproduct of the cache mapping algorithm. A conflict miss occurs because too many active blocks are mapped to the same cache set.

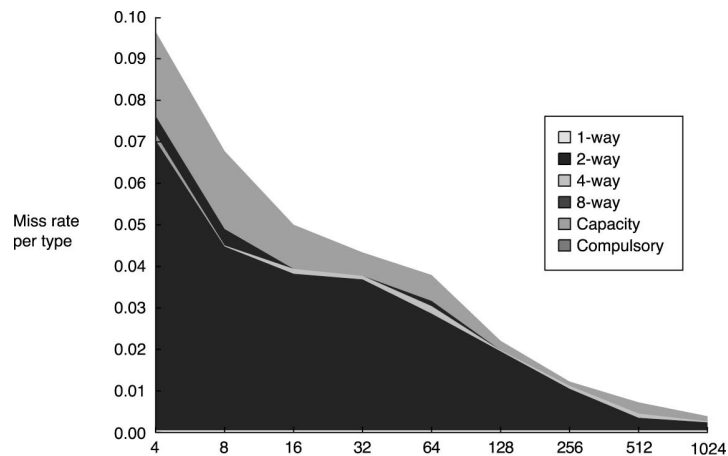
How To Measure

Misses in infinite cache

Non-compulsory misses in size X fully associative cache

Non-compulsory, non-capacity misses

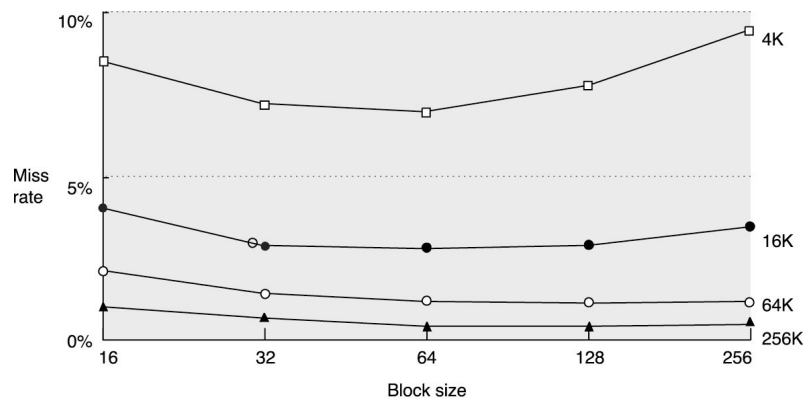
3Cs Absolute Miss Rate



How To Reduce Misses?

- Compulsory Misses?
- Capacity Misses?
- Conflict Misses?
- What can the compiler do?

Reduce Misses via Larger Block Size



- 16K cache, miss penalty for 16-byte block = 42, 32-byte is 44, 64-byte is 48. Miss rates are 3.94, 2.87, and 2.64%?

CSE 240A

Dean Tullsen

Reduce Misses via Higher Associativity

- 2:1 Cache Rule:
 - MR of DM cache size N - MR of 2-way cache size N/2
- Beware: Execution time is only final measure!
 - Will Clock Cycle time increase?
 - Hill [1988] suggested hit time external cache +10%, internal + 2% for 2-way vs. 1-way

CSE 240A

Dean Tullsen

Example: Avg. Memory Access Time vs. Miss Rate

- Example: assume CT = 1.10 for 2-way, 1.12 for 4-way, 1.14 for 8-way vs. CT direct mapped

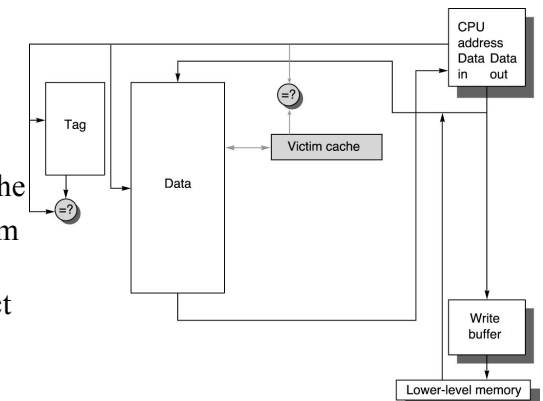
| Cache Size (KB) | Associativity | | | |
|-----------------|---------------|-------|-------|-------------|
| | 1-way | 2-way | 4-way | 8-way |
| 1 | 7.65 | 6.60 | 6.22 | 5.44 |
| 2 | 5.90 | 4.90 | 4.62 | 4.09 |
| 4 | 4.60 | 3.95 | 3.57 | 3.19 |
| 8 | 3.30 | 3.00 | 2.87 | 2.59 |
| 16 | 2.45 | 2.20 | 2.12 | 2.04 |
| 32 | 2.00 | 1.80 | 1.77 | 1.79 |
| 64 | 1.70 | 1.60 | 1.57 | 1.59 |
| 128 | 1.50 | 1.45 | 1.42 | 1.44 |

CSE 240A

Dean Tullsen

Reducing Misses by emulating associativity: Victim Cache

- HR of associative + access time of direct mapped?
- Add buffer to hold data recently discarded from cache
- Jouppi [1990]: 4-entry victim cache removed 20% to 95% of conflicts for a 4 KB direct mapped data cache



CSE 240A

Dean Tullsen

Reducing Misses by emulating associativity: Pseudo-Associativity

- Combines fast hit time of Direct Mapped and the lower conflict misses of a 2-way SA cache.
- Divide cache: on a miss, check other half of cache to see if there, if so have a pseudo-hit (slow hit)



- Drawback: CPU pipeline is hard if hit can take 1 or 2 cycles
 - Better for caches not tied directly to processor

Reducing Misses by HW Prefetching of Instruction & Data

- E.g., Instruction Prefetching
 - Alpha 21064 fetches 2 blocks on a miss
 - Extra block placed in stream buffer
 - On miss check stream buffer
- Works with data blocks too:
 - Jouppi [1990] 1 data stream buffer got 25% misses from 4KB cache; 4 streams got 43%
 - Palacharla & Kessler [1994] for scientific programs for 8 streams got 50% to 70% of misses from 2 64KB, 4-way set associative caches
- Prefetching relies on extra memory bandwidth that can be used without penalty

Reducing Misses by SW Prefetching Data

- Data Prefetch
 - Load data into register (HP PA-RISC, IA64, Tera)
 - Cache Prefetch: load into cache (MIPS IV, PowerPC, SPARC)
 - Special prefetching instructions cannot cause faults; a form of speculative execution
- Issuing Prefetch Instructions takes time
 - Is cost of prefetch issues < savings in reduced misses?

Reducing Misses by Various Compiler Optimizations

- Instructions
 - Reorder procedures in memory so as to reduce misses
 - Profiling to look at conflicts
 - McFarling [1989] reduced cache misses by 75% on 8KB direct mapped cache with 4 byte blocks
- Data
 - **Merging Arrays**: improve spatial locality by single array of compound elements vs. 2 arrays
 - **Loop Interchange**: change nesting of loops to access data in order stored in memory
 - **Loop Fusion**: Combine 2 independent loops that have same looping and some variables overlap
 - **Blocking**: Improve temporal locality by accessing “blocks” of data repeatedly vs. going down whole columns or rows

Merging Arrays Example

```
/* Before */
int val[SIZE];
int key[SIZE];

/* After */
struct merge {
    int val;
    int key;
};
struct merge merged_array[SIZE];
```

Reducing conflicts between val & key

CSE 240A

Dean Tullsen

Loop Interchange Example

```
/* Before */
for (k = 0; k < 100; k = k+1)
    for (j = 0; j < 100; j = j+1)
        for (i = 0; i < 5000; i = i+1)
            x[i][j] = 2 * x[i][j];

/* After */
for (k = 0; k < 100; k = k+1)
    for (i = 0; i < 5000; i = i+1)
        for (j = 0; j < 100; j = j+1)
            x[i][j] = 2 * x[i][j];
```

Sequential accesses instead of striding through memory every 100 words

CSE 240A

Dean Tullsen

Loop Fusion Example

```
/* Before */
for (i = 0; i < N; i = i+1)
    for (j = 0; j < N; j = j+1)
        a[i][j] = 1/b[i][j] * c[i][j];
for (i = 0; i < N; i = i+1)
    for (j = 0; j < N; j = j+1)
        d[i][j] = a[i][j] + c[i][j];

/* After */
for (i = 0; i < N; i = i+1)
    for (j = 0; j < N; j = j+1)
    {
        a[i][j] = 1/b[i][j] * c[i][j];
        d[i][j] = a[i][j] + c[i][j];
    }
```

2 misses per access to a & c vs. one miss per access

CSE 240A

Dean Tullsen

Blocking Example

```
/* Before */
for (i = 0; i < N; i = i+1)
    for (j = 0; j < N; j = j+1)
    {
        r = 0;
        for (k = 0; k < N; k = k+1) {
            r = r + y[i][k] * z[k][j];
        }
        x[i][j] = r;
    }
```

- Two Inner Loops:
 - Read all NxN elements of z[]
 - Read N elements of 1 row of y[] repeatedly
 - Write N elements of 1 row of x[]
- Capacity Misses a function of N & Cache Size:
 - worst case $\Rightarrow 2N^3 + N^2$.
- Idea: compute on BxB submatrix that fits in cache

CSE 240A

Dean Tullsen

Blocking Example

```
/* After */
for (jj = 0; jj < N; jj = jj+B)
for (kk = 0; kk < N; kk = kk+B)
for (i = 0; i < N; i = i+1)
  for (j = jj; j < min(jj+B-1,N); j = j+1)
    {r = 0;
     for (k = kk; k < min(kk+B-1,N); k = k+1) {
       r = r + y[i][k]*z[k][j];
       x[i][j] = x[i][j] + r;
     };
  };
```

- Capacity Misses from $2N^3 + N^2$ to $2N^3/B + N^2$
- B called *Blocking Factor*
- Conflict Misses Are Not As Easy...

CSE 240A

Dean Tullsen

Key Points

$$CPUtime = IC \times \left(CPI_{Execution} + \frac{Memory\ accesses}{Instruction} \times Miss\ rate \times Miss\ penalty \right) \times Clock\ cycle\ time$$

- 3 Cs: Compulsory, Capacity, Conflict Misses
- Reducing Miss Rate
 - 1. Reduce Misses via Larger Block Size
 - 2. Reduce Misses via Higher Associativity
 - 3. Reducing Misses via Victim Cache
 - 4. Reducing Misses via Pseudo-Associativity
 - 5. Reducing Misses by HW Prefetching Instr, Data
 - 6. Reducing Misses by SW Prefetching Data
 - 7. Reducing Misses by Compiler Optimizations
- Remember danger of concentrating on just one parameter when evaluating performance
- Next: reducing Miss penalty

CSE 240A

Dean Tullsen