The hardware cache

Exploiting temporal and spatial locality

Introduction

Locality

Cache organization

Conclusion

Temporal locality

- The reuse of data within some (usually short) time duration
- If the processor accesses data, it will likely access it again (soon)

variable	+ 0x0	+ 0x4	+ 0x8	+ 0xC
a	2	7	3	5
	9	12	14	16
	18	23		

```
for(int i = 0; i < n - 1; i++) {
    int min_index = i;
    for(int j = i + 1; j < n; j++) {
        if(a[j] < a[min_idx]) {
            min_index = j;
        }
    }

    int temp = a[min_index];
    a[min_index] = a[i];
    a[i] = temp;
}
```

Spatial locality

- The use of data that are located closely together
- If the processor accesses data, it will likely access other nearby memory locations

variable	+ 0x0	+ 0x4	+ 0x8	+ 0xC
a	2	7	3	5
	9	12	14	16
	18	23		

```
for(int i = 0; i < n - 1; i++) {
    int min_index = i;
    for(int j = i + 1; j < n; j++) {
        if(a[j] < a[min_idx]) {
            min_index = j;
        }
    }

    int temp = a[min_index];
    a[min_index] = a[i];
    a[i] = temp;
}
```

The cache block

- A *cache block* is a group of words
- The number of words in a cache block is the *block size* (b)
- The cache capacity (C) contains $B = C/b$ blocks

variable	+ 0x0	+ 0x4	+ 0x8	+ 0xC
a	2	7	3	5
	9	12	14	16
	18	23		

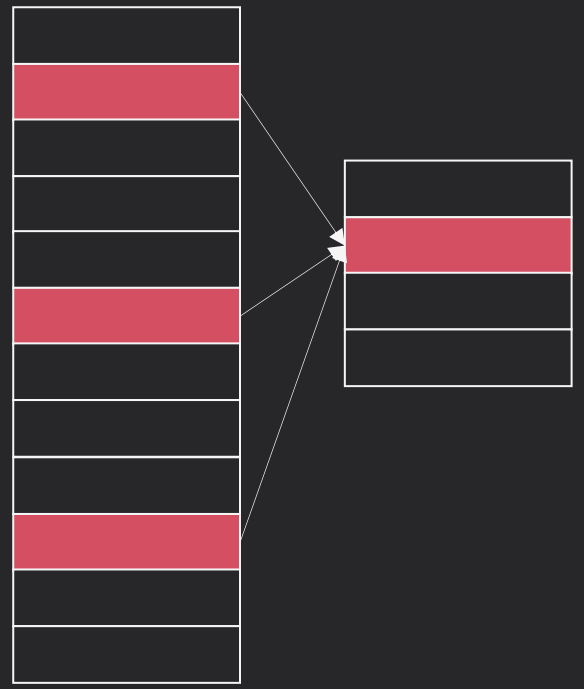
```
for(int i = 0; i < n - 1; i++) {
    int min_index = i;
    for(int j = i + 1; j < n; j++) {
        if(a[j] < a[min_idx]) {
            min_index = j;
        }
    }

    int temp = a[min_index];
    a[min_index] = a[i];
    a[i] = temp;
}
```

Cache organization

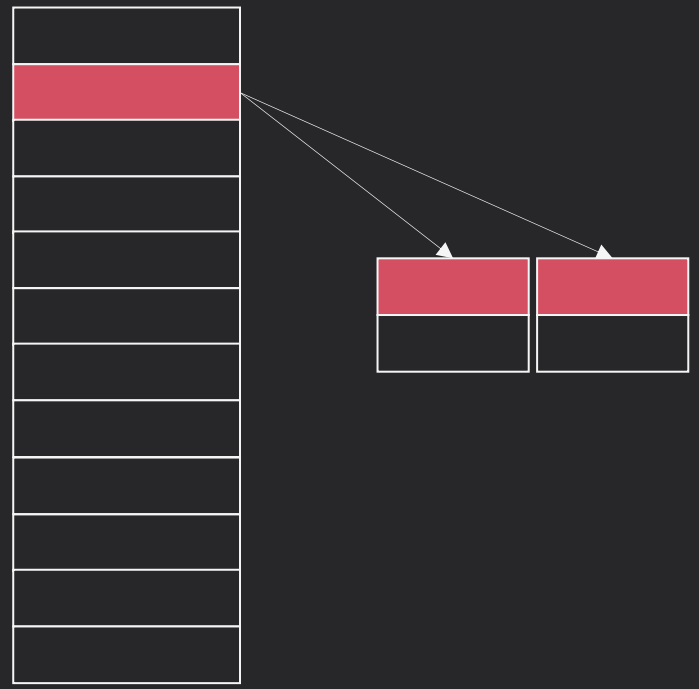
- Caches are organized into (S) *sets*. A set consists of one or more blocks
- Each memory address maps to one set in the cache

Direct-mapped cache



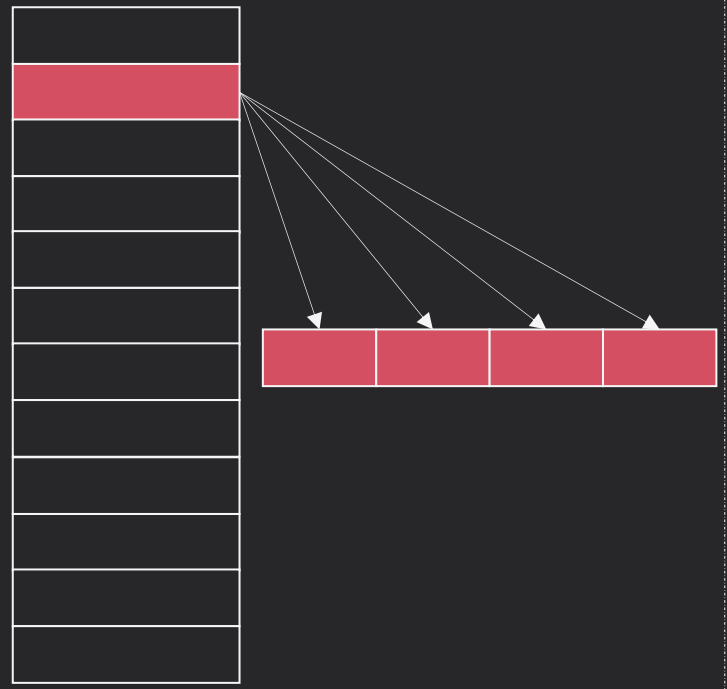
Each set contains one block ($S = B$).

N-way set-associative cache



Each set contains N blocks ($S = {}^{B}/_{N}$).
Data from memory address goes into *any* block in its corresponding set.

Fully associative cache

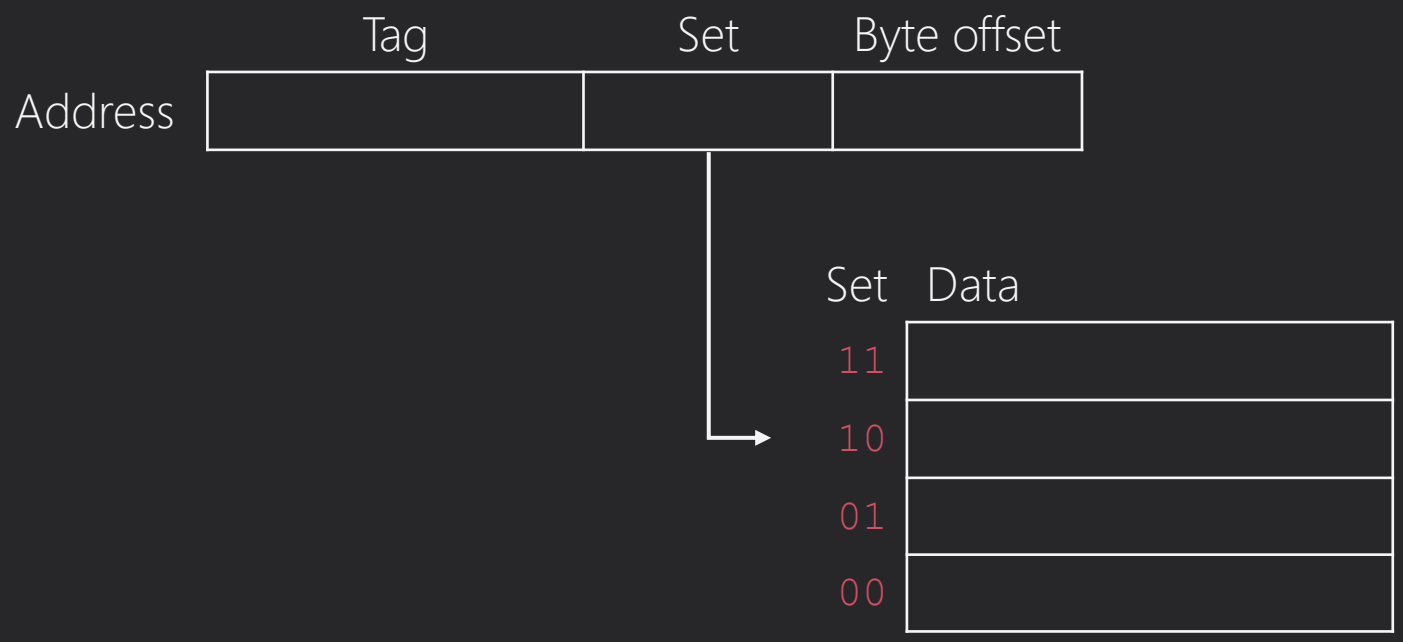


There is only 1 set.
Data can go in any of the B blocks in the set.

Direct mapped cache

- Addresses are word-aligned
- Example of direct mapped cache with B = 4
- Mapping: $Set = Address \bmod 4$

Address	Data
00…00 10 0000	mem[0x20]
00…00 01 1100	mem[0x1C]
00…00 01 1000	mem[0x18]
00…00 01 0100	mem[0x14]
00…00 01 0000	mem[0x10]
00…00 00 1100	mem[0x0C]
00…00 00 1000	mem[0x08]
00…00 00 0100	mem[0x04]
00…00 00 0000	mem[0x00]



The tag bits

- Many addresses map to a single set
- The *tag* identifies the memory address of the data contained in the set
  - Compare tag from address with tag in cache

Address	Data
00…00 10 0000	mem[0x20]
00…00 01 1100	mem[0x1C]
00…00 01 1000	mem[0x18]
00…00 01 0100	mem[0x14]
00…00 01 0000	mem[0x10]
00…00 00 1100	mem[0x0C]
00…00 00 1000	mem[0x08]
00…00 00 0100	mem[0x04]
00…00 00 0000	mem[0x00]

Set	Tag	Data
11		
10		
01		
00		

The valid bits

- The *valid bit* indicates whether the data contents are meaningful
  - e.g., valid bits are all 0 when a computer turns on

Address	Data
00…00 10 0000	mem[0x20]
00…00 01 1100	mem[0x1C]
00…00 01 1000	mem[0x18]
00…00 01 0100	mem[0x14]
00…00 01 0000	mem[0x10]
00…00 00 1100	mem[0x0C]
00…00 00 1000	mem[0x08]
00…00 00 0100	mem[0x04]
00…00 00 0000	mem[0x00]

Set	V	Tag	Data
11			
10			
01			
00			

Conflicts

- Two accesses that map to the same cache block cause a *conflict*
  - The more recent access *evicts* the previous one
- Caches with a *degree of associativity* (N) can reduce conflicts
  - Direct mapped cache has N = 1
  - When N >= 2, addresses that map to the same set can choose one of N blocks
- Trade-offs with associativity
  - Higher hardware overhead
  - What if all blocks are being used? Need a replacement policy

Conclusion

Recapping the important points

A summary of cache organizations

- A *cache block* is a group of words
  - The number of words in a cache block is the *block size* (b)
  - The cache capacity (C) contains $B = C/b$ blocks
- Caches are organized into (S) *sets*.
  - A set consists of one or more blocks
  - Each memory address maps to one set in the cache
- Caches with a *degree of associativity* (N) can reduce conflicts
  - But this has some overhead and requires a replacement policy

Organization	N	S
Direct mapped	1	B
N-way set associative	1 < N < B	B / N
Fully associative	B	1