Department of Electrical and Computer Engineering

The University of Texas at Austin

ECE 460N/382N.1, Spring 2025
Problem Set 4
Due: November 11th, before class
Yale N. Patt, Instructor
Luke Mason, Roy Mor, Jenna May, Margaret Lee, Rathna Sivakumar

Instructions

You are encouraged to work on the problem set in groups and turn in one problem set for the entire group. The problem sets are to be submitted on Gradescope. Only one student should submit the problem set on behalf of the group, but everyone should create a gradescope account and be tagged on the homework. **You will need to assign your questions to pages on gradescope. Each problem you fail to do this will receive a 3 point deduction.**

You will need to refer to the assembly language handout and the LC-3b ISA, microarchitecture, and state diagram documents on the course website.

Cache Memory Questions:

Problem 1

1) Direct mapped would be better for the following case : If 9 addresses were cycled, each with different sets (other than the 9th address which would need to share a set with one of the other addresses), in the direct mapped cache, only two of the addresses would ever get evicted. However, for the fully associative cache, since there are only 8 spots, by the time we cycled back to an address, it would have just gotten kicked out (by means of LRU). In this case, we would have 7/9 hits for the direct mapped cache (from the second cycle onwards), while we would have 0 hits for the fully associative cache.
2) Fully associative would be better for the following case : 4 addresses, each with different tag and same set are cycled. For the direct mapped cases, there are 0 hits since all 4 addresses share the same set. For the fully associative case, there are 4 hits from the second cycle onwards since none of the addresses leave the cache.
3) Tradeoffs : Less LRU bits for the direct mapped / 2 way vs more LRU bits for the 4 way / fully associative. More hardware needed to read and compare tag bits for the 4 way and fully associative vs 2 way and direct mapped. More associativity usually means more hits and less evictions for the 4 way vs 2 way (for the average workload, not the specific cases above).

Problem 2

An LC-3b system with a 384B cache using an LRU (least recently used) replacement scheme has the following access pattern (the cache is initially empty):

```
x3000 MISS
x3006 HIT
x3008 MISS
x3104 MISS
x3180 MISS
x300A HIT
x3142 MISS
x3200 MISS
x3006 MISS
x3140 HIT
x3100 MISS
x3186 HIT
```

a. What is the block size of the cache?
8 Byte block size

b. What is the associativity of the cache? If set-associative, how many ways and sets are there?
3-Way Set Associative, 16 sets

c. What is the address mapping of the cache? (tag bits, index bits, offset bits)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Tag	Tag	Tag	Tag	Tag	Tag	Tag	Tag	Tag	Idx	Idx	Idx	Idx	Off	Off	Off

Problem 3

A computer has an 8KB write-through cache. Each cache block is 64 bits, the cache is 4-way set associative and uses a victim/next-victim pair of bits for each block for its replacement policy.
Assume a 24-bit address space and byte-addressable memory. How big (in bits) is the tag store?

An 8KB cache size with a 8B line size, in a 4-way set associative cache means there are 8KB ÷ (4 × 8B) = 256 sets in the cache.
Since there are 256 or 2

8 sets, 8 bits are required to index into the correct set. Since there are

8B or 2
3 bytes in a cache line, 3 bits are required to find the correct byte within a block. Given a 24-bit address space, this leaves 24 - 8 - 3 = 13 bits left over for the tag store. Additionally, the tag store must hold 2 bits for the V/NV replacement policy and 1 valid bit. This means each cache line must have a 16-bit tag store associated with it. 2B of tag store times (256 × 4) cache lines in the cache means that the tag store, in total takes up 2048 bytes, which is 16384 bits

Problem 4

An LC-3b system ships with a two-way set associative, write back cache with perfect LRU replacement. The tag store requires a total of 4352 bits of storage. What is the block size of the cache? Please show all your work.

Hint: $4352 = 2^{12} + 2^{8}$

The size of the tag store is 2
12 + 2
8
. We know that the size of the tag store can be given as the

product of number of sets and number of bits per set.
We also know that address space is 16-bits. Hence, tag + index + bib = 16
In order to find bib, we need to find index and tag. The following bits are necessary for each set of the tag store:
- 1 bit for LRU (remember: you only need one bit per set for a 2-way associative cache)
- 2 valid bits
- 2 dirty bits
- 2 × tag tag bits

Total number of bits per set is 5 + 2 × tag. An important conclusion that can be drawn is that number of bits per set will always be an odd number.
We will also use the fact that number of sets is always a power of 2 (since it is indexed using an integer number of bits). Given the size of the tag store, index has to be a number less than 8 (the size of the tag store is indivisible by 2

9 or greater).
The only value of index that fits both criterion is 8. Therefore:
Number of sets = 2
8 = 256
Bits per row = 4352 ÷ 256 = 17
17 = 5 + 2 × tag ⇒ tag = 6
6 + 8 + bib = 16 ⇒ bib = 2
Hence, the cache block size is 4 bytes

Problem 5

Based on Hamacher et al., p. 255, question 5.18. You are working with a computer that has a first level cache that we call L1 and a second level cache that we call L2. Use the following information to answer the questions.

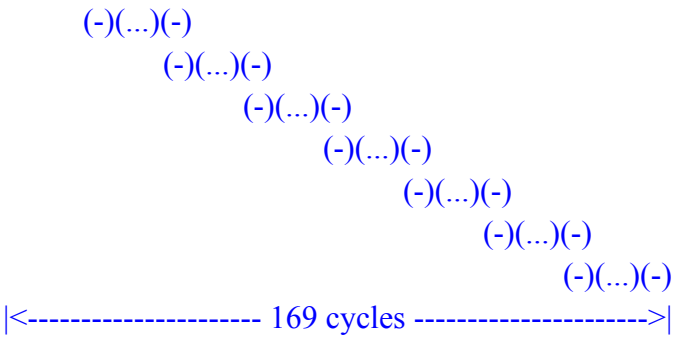
- The L1 hit rate is 0.95 for instruction references and 0.90 for data references.
- The L2 hit rate is 0.85 for instruction references and 0.75 for data references.
- 30% of all instructions are loads and stores.

● The size of each cache block is 8 words.
● The time needed to access a cache block in L1 is 1 cycle and the time needed to access a cache block in L2 is 6 cycles.

● The accesses to the caches and memory are done sequentially. If there is a miss in the L1 and a hit in the L2 then the total latency is 7 cycles.
● Memory is accessed only if there is a miss in both caches.
● The width of the memory bus is one word.
● It takes one clock cycle to send an address to main memory.
● It takes 20 cycles to access the main memory.
● It takes one cycle to send one word from the memory to the processor. Thus the total latency to get a word from memory to the processor is 22 cycles.
● The bus allows sending a new address to memory in the same cycle that data is sent from memory to the processor.
● Assume the data is accessible to the processor only AFTER the whole cache block has been brought in from the memory, and buffered on the processor chip. The processor can then access the data independent of and during the cache fill.

1. What is the average access time per instruction (assume no interleaving)?
2. What is the average access time per instruction if the main memory is 4-way interleaved?
3. What is the average access time per instruction if the main memory is 8-way interleaved?
4. What is the improvement obtained with interleaving?

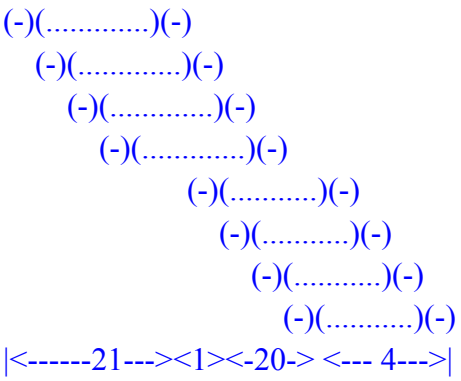
average_access_time_per_instruction = 1 × instruction_access_time + 0.3 × data_access_time
We can use the following equations in both part a and part b to compute the instruction and data access time.
instruction_access_time = 1 + 0.05 × (6 + 0.15 × mem_latency)
data_access_time = 1 + 0.10 × (6 + 0.25 × mem_latency)
1. We need to calculate the memory latency. Since each cache block is 8 words, it will take 8 accesses to get a cache block from memory.



Using the equations, we get 2.5675 cycles for instruction_access_time and 5.825 for data_access_time.

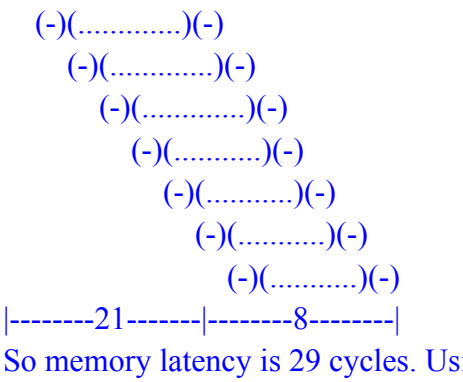
The average latency is 2.5675 + 0.3 × 5.825 = 4.315 cycles

2. We again need to compute the memory latency:



So the mem_latency is 46 cycles. Again using the equations, the instruction_access_time is 1.645 and data_acces_time is 2.75.
The average latency is 1.645 + 2.75 × 0.3 = 2.47 cycles

3. We need to compute memory latency for 8-way interleaving



So memory latency is 29 cycles. Using the equations, we get the instruction_access_time as 1.517 and data_acces_time as 2.325.
The average latency is 1.517 + .3 * 2.325 = 2.215

4. For the 4-way interleaved memory, the improvement is (4.315 - 2.47) ÷ 4.315 = 0.4276
Therefore, the average latency improves by 42.76%
For the 8-way interleaved memory, the improvement is (4.315 - 2.215) ÷ 4.315 = 0.4867
Therefore, the average latency improves by 48.67%

Problem 6

Hamacher, pg.255, question 5.13. A byte-addressable computer has a small data cache capable of holding eight 32-bit words. Each cache block consists of one 32-bit word. When a

given program is executed, the processor reads data from the following sequence of hex addresses:
200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4
This pattern is repeated four times.
1. Show the contents of the cache at the end of each pass throughout this loop if a direct-mapped cache is used. Compute the hit rate for this example. Assume that the cache is initially empty.
2. Repeat part (a) for a fully-associative cache that uses the LRU-replacement algorithm.
3. Repeat part (a) for a four-way set-associative cache that uses the LRU replacement algorithm.

1. The address is divided into 3 portions:
- address[1:0] (2 bits) for the byte in the block
- address [4:2] (3 bits) for the cache index
- address[11:5] (7 bits) for the tag

The contents of the cache at the end of each pass are the same. They are shown below:

Valid	Tag	Data (addresses are written inside each byte)
1	0010	000 203 202 201 200
1	0010	000 207 206 205 204
1	0010	000 20B 20A 209 208
1	0010	010 24F 24E 24D 24C
1	0010	111 2F3 2F2 2F1 2F0
1	0010	111 2F7 2F6 2F5 2F4
1	0010	000 21B 21A 219 218
1	0010	000 21F 21E 21D 21C

The hit/miss information for each pass is shown below:

Reference	200	204	208	20C	2F4	2F0	200	204	218	21C	24C	2F4
Pass 1:	M	M	M	M	M	M	H	H	M	M	M	H
Pass 2:	H	H	H	M	H	H	H	H	H	H	M	H
Pass 3:	H	H	H	M	H	H	H	H	H	H	M	H
Pass 4:	H	H	H	M	H	H	H	H	H	H	M	H

Hit rate is 33/48
2. The address is divided into two: 2 bits for identifying the byte in the block, 10 bits for the tag. No bits are needed for cache index. The following table shows the contents of the cache at the end of each pass (Valid bits and Tags are ignored, they should be obvious. Starting addresses of blocks in the cache are provided):

Pass	Way 0
Data	
Way 1	

Data

Way 2
Data

Way 3
Data

Way 4
Data

Way 5
Data

Way 6
Data

Way 7
Data
1 200 204 24C 20C 2F4 2F0 218 21C
2 200 204 21C 24C 2F4 20C 2F0 218
3 200 204 218 21C 2F4 24C 20C 2F0
4 200 204 2F0 218 2F4 21C 24C 20C
The hit/miss information for each pass is shown below:
Reference: 200 204 208 20C 2F4 2F0 200 204 218 21C 24C 2F4
Pass 1: M M M M M M H H M M M H
Pass 2: H H M M H M H H M M M H
Pass 3: H H M M H M H H M M M H
Pass 4: H H M M H M H H M M M H
Hit rate is 21/48
3. The address is divided into 3 portions: 2 bits for identifying the byte in the block, 1 bit for the cache index, 9 bits for the tag. The contents of the cache at the end of Pass 1:
Way 0 Way 1 Way 2 Way 3
V Tag Data V Tag Data V Tag Data V Tag Data
1 0010 0000
0

203-20
0
1 0010 0000
1

20B-20
8
1 0010 1111

0

2F3-2F
0
1 0010 0001
1

21B-21
8

1 0010 0000
0

207-20
4
1 0010 0100
1

24F-24
C
1 0010 1111
0

2F7-2F
4
1 0010 0001
1

21F-21
C

The hit/miss information for each pass is shown below:

Reference: 200 204 208 20C 2F4 2F0 200 204 218 21C 24C 2F4
Pass 1: M M M M M M H H M M M H
Pass 2: H H H M H H H H H M M H
Pass 3: H H H M H H H H H M M H
Pass 4: H H H M H H H H H M M H
Contents of the second set of the cache (index equals 1) after pass 2, 3, and 4 (the first set remains the same):
Pass Way 0 Way 1 Way 2 Way 3
V Tag Data V Tag Data V Tag Data V Tag Data
2 1 0010 0000
0

207-20
4
1 0010 0001
1

21F-21
C
1 0010 1111
0

2F7-2F
4
1 0010 0100
1

24F-24
C

3 1 0010 0000
0

207-20
4
1 0010 0100
1

24F-24
C
1 0010 1111
0

2F7-2F
4
1 0010 0001
1

21F-21
C

4 1 0010 0000
0

207-20
4
1 0010 0001

1

21F-21
C
1 0010 1111
0

2F7-2F
4
1 0010 0100
1

24F-24
C

Hit Rate is 30/48

Problem 7

Below, we have given you four different sequences of addresses generated by a program running on a processor with a data cache. Cache hit ratio for each sequence is also shown below. Assuming that the cache is initially empty at the beginning of each sequence, find out the following parameters of the processor's data cache:

- Associativity (1, 2, or 4 ways)
- Block size (1, 2, 4, 8, 16, or 32 bytes)
- Total cache size (256B, or 512B)
- Replacement policy (LRU or FIFO)

Assumptions: all memory accesses are one byte accesses. All addresses are byte addresses.

Number	Address Sequence	Hit Ratio
1	0, 2, 4, 8, 16 ,32	0.33
2	0, 512, 1024, 1536, 2048, 1536, 1024, 512, 0	0.33
3	0, 64, 128, 256, 512, 256, 128, 64, 0	0.33
4	0, 512, 1024, 0, 1536, 0, 2048, 512	0.25

Block Size
The following table lists hit ratios for different cache block sizes given the access sequence 1 (0, 2, 4, 8, 16, 32):

Block Size Hit Ratio
1B 0/6
2B 0/6
4B 1/6
8B 2/6
16B 3/6
32B 4/6
Since the hit ratio is reported as 0.33 for this sequence, the block size must be 8 bytes. Therefore, the accesses look like this:
Address 0 2 4 8 16 32
Hit/Miss M H H M M M
Associativity
Notice that all of the addresses of sequence 2 (0, 512, 1024, 1536, 2048, 1536, 1024, 512, 0) are multiples of 512. This means that they all map to the same set, since the size of the cache can only be 256 or 512 bytes. Therefore, the hit ratio would be 0/9 in case of a direct-mapped cache. In case of a 2-way cache, the accesses would behave as follows:
Address 0 512 1024 1536 2048 1536 1024 512 0
Hit/Miss M M M M M H M M M
The resulting hit rate would be 1/9. Similarly, for 4-way cache:
Address 0 512 1024 1536 2048 1536 1024 512 0
Hit/Miss M M M M M H H H M
The resulting hit rate would be 3/9, and thus the cache is 4-way set associative.
Cache Size

If the cache size were 256B, there would be 3 index bits and all of the addresses in sequence 3 (0, 64, 128, 256, 512, 256, 128, 64, 0) would map to the same set:
Address 0 64 128 256 512 256 128 64 0
Hit/Miss M M M M M H H H M
The resulting hit ratio would be 3/9, and thus the cache size is 256B. For completeness, here's how the accesses would look like in case of a 512B cache (4 index bits):
Address 0 64 128 256 512 256 128 64 0
Hit/Miss M M M M M H H H H
Replacement Policy
In case of the FIFO replacement policy, the accesses of sequence 3 (0, 512, 1024, 0, 1536, 0, 2048, 512) would look as follows:
Address 0 512 1024 0 1536 0 2048 512
Hit/Miss M M M H M H M H
The resulting hit ration would be 3/8. However, in case of the LRU replacement policy the hit rate would be 0.25:
0 512 1024 0 1536 0 2048 512
M M M H M H M M
Thus the replacement policy is LRU.

Arithmetic Questions:

Problem 8

Binary-coded Decimal:

An integer, expressed as a 16-bit BCD is 0000 0100 0110 0000.

a. What is its decimal representation?

460

b. What is its 16-bit 2's complement representation?

x01CC

c. A register filled with xXXXX is required to add two BCD values in hardware, where xXXXX represents a 16-bit value. What is xXXXX and why?

xXXXX is x6666 to allow for BCD arithmetic. The addition of x6 means that a carry is created when xA (#10) is the sum, not when x10 (#16) is the sum.

Problem 9 (not be graded)

Using Booth's algorithm, we want to perform an integer multiplication, which is 79 x 30. Since both given values fit within 8 bits, we need at most 4 iterations to complete the multiplication.

1. Let multiplicand be 79 and multiplier be 30. Show the values of the registers MCAND and MULTIPLIER at the end of each iteration.

	MCAND	MULTIPLIER
1	01001111	00011110
2	01001111	00000111
3	01001111	00000001
4	01001111	00000000

2. Now let multiplicand be 30 and multiplier be 79. Show the values of the registers MCAND and MULTIPLIER at the end of each iteration.

	MCAND	MULTIPLIER
1	00011110	01001111
2	00011110	00010111
3	00011110	00000101
4	00011110	00000001

3. Express Booth's Algorithm for 79 × 30 in a single equation using decimal values. (mcand =79 and multiplier=30)

79 x (16 + 4 x (4 - 1) + 2) or 79 x (16 + 8 + 4 + 2)