Questions:

4.1 What are the differences among sequential access, direct access, and random access?

- **Sequential access**:
  Memory is organized into units of data, called records. Access must be made in a specific linear sequence.
- **Direct access**:
  Individual blocks or records have a unique address based on physical location. Access is accomplished by direct access to reach a general vicinity plus sequential searching, counting, or waiting to reach the final location.
- **Random access**:
  Each addressable location in memory has a unique, physically wired-in addressing mechanism. The time to access a given location is independent of the sequence of prior accesses and is constant.

4.2 What is the general relationship among access time, memory cost, and capacity?

- Faster access time, greater cost per bit; greater capacity, smaller cost per bit; greater capacity, slower access time.

4.3 How does the principle of locality relate to the use of multiple memory levels?

- It is possible to organize data across a memory hierarchy such that the percentage of accesses to each successively lower level is substantially less than that of the level above. Because memory references tend to cluster, the data in the higher level memory need not change very often to satisfy memory access requests.

4.4 What are the differences among direct mapping, associative mapping, and set-associative mapping?

- **Direct mapping:** maps each block of main memory into only one possible cache line.
- **Associative mapping:** permits each main memory block to be loaded into any line of the cache.
- **Set-associative mapping:** the cache is divided into a number of sets of cache lines; each main memory block can be mapped into any line in a particular set.

4.5 For a direct-mapped cache, a main memory address is viewed as consisting of three fields. List and define the three fields.

- The three fields are [**Tag , Line , Offset**]
- **Tag:** identifies one of the blocks that can fit into that line.
- **Line:** identifies one of the lines of the cache.
- **Offset (word):** identifies a unique word or byte within a block of main memory.

4.6 For an associative cache, a main memory address is viewed as consisting of two fields. List and defines the two fields.

- **Tag:** uniquely identifies a block of main memory.
- **Offset (word):** identifies a unique word or byte within a block of main memory

4.7 For a set-associative cache, a main memory address is viewed as consisting of three fields. List and define the three fields.

- The three fields are [**Tag , Set , Offset**]
- **Tag:** identifies one of the blocks that can fit into that set.
- **Set:** specify one of the blocks of main memory.
- **Offset (word):** identifies a unique word or byte within a block of main memory.

4.8 What is the distinction between spatial locality and temporal locality?

- **Spatial locality** refers to the tendency of execution to involve a number of memory locations that are clustered (closer to each other).
- **Temporal locality** refers to the tendency for a processor to access memory locations that have been used recently.

4.9 In general, what are the strategies for exploiting spatial locality and temporal locality?

- **Spatial locality** is generally exploited by using larger cache blocks and by incorporating prefetching mechanisms (fetching items of anticipated use) into the cache control logic.
- **Temporal locality** is exploited by keeping recently used instruction and data values in cache memory and by exploiting a cache hierarchy.

Problems:

4.1 A set-associative cache consists of 64 lines, or slots, divided into four-line sets. Main memory contains 4K blocks of 128 words each. Show the format of main memory addresses.

Answer:

Type of cache = set-associative

Number of lines in Cache = 64 lines

Number of lines in a set = 4 lines/set

Number of blocks in main memory = 4k blocks

Number of words in a block = 128 words

Memory size in words = Number of blocks in main memory * Number of words in a block

= 4 * 2^{10} * 128 = 524288 words = 2^{19} words

Number of bits in the address of one word = $log_2(\text{Memory size in words}) = log_2(2^{19})$ = 19 bits

Number of sets in cache = $\frac{\text{Number of lines in Cache}}{\text{Number of lines in a set}} = \frac{64}{4}$ = 16 set

Number of bits for set field = $log_2(\text{Number of sets in cache}) = log_2(16) = 4$ bits

Number of bits in offset (word) field = $log_2(\text{Number of words in a line}) = log_2(128) = 7$ bits

Number of bits in tag field = Number of bits in the address – (Number of bits for set field + Number of bits in offset (word) field)

= 19 – (4 + 7) = 8 bits

	Tag	Set	Offset (word)
Address(19 bits total)	8 bits	4 bits	7 bits

4.2 A two-way set-associative cache has lines of 16 bytes and a total size of 8 Kbytes. The 64-Mbyte main memory is byte addressable. Show the format of main memory addresses.

Type of cache = set-associative

Number of lines in a set = 2 lines/set ➔ (two-way set-associative)

Size of line = 16 bytes

Total size of Cache = 8 Kbytes

Total size of Main memory= 64 Mbytes

Number of bits in the address of one word = $log_2(\text{Memory size in bytes}) = log_2(64 * 2^{20})$ = 26 bits

Number of lines in cache = $\frac{\text{Total size of Cache}}{\text{Size of line}} = \frac{8 * 2^{10}}{16}$ = 512 lines

Number of sets in cache = $\frac{\text{Number of lines in Cache}}{\text{Number of lines in a set}} = \frac{512}{2}$ = 256 set

Number of bits for set field = $log_2(\text{Number of sets in cache}) = log_2(256) = 8$ bits

Number of bits in offset (byte) field = $log_2(\text{Number of bytes in a line}) = log_2(16) = 4$ bits

Number of bits in tag field = Number of bits in the address – (Number of bits for set field + Number of bits in offset (word) field)

= 26 – (8 + 4) = 14 bits

	Tag	Set	Offset (byte)
Address(26 bits total)	14 bits	8 bits	4 bits

4.3 For the hexadecimal main memory addresses 111111, 666666, BBBBBB, show the following information, in hexadecimal format:

a. Tag, Line, and Word values for a direct-mapped cache, using the format of Figure 4.10

b. Tag and Word values for an associative cache, using the format of Figure 4.12

c. Tag, Set, and Word values for a two-way set-associative cache, using the format of Figure 4.15

Answer:

A – from figure 4.10:

	Tag	Line	Offset (word)
Address(24 bits total)	8 bits	14 bits	2 bits

To get values of each field , change address to binary format then cut to pieces:

Address	Tag (8)	Line (14)	Offset (2)	Tag(in hex)	Line(in hex)	Offset (in hex)
0x111111	0001 0001	0001 0001 0001 00	01	11	444	1
0x666666	0110 0110	0110 0110 0110 01	10	66	1999	2
0xBBBBBB	1011 1011	1011 1011 1011 10	11	BB	2EEE	3

B – from figure 4.12:

	Tag	Offset (word)
Address(24 bits total)	22 bits	2 bits

To get values of each field , change address to binary format then cut to pieces:

Address	Tag (22)	Offset (2)	Tag(in hex)	Offset (in hex)
0x111111	0001 0001 0001 0001 0001 00	01	44444	1
0x666666	0110 0110 0110 0110 0110 01	10	19999	2
0xBBBBBB	1011 1011 1011 1011 1011 10	11	2EEEE	3

C – from figure 4.15:

	Tag	Set	Offset (word)
Address(24 bits total)	9 bits	13 bits	2 bits

To get values of each field , change address to binary format then cut to pieces:

Address	Tag (9)	Set (13)	Offset (2)	Tag(in hex)	Line(in hex)	Offset (in hex)
0x111111	0001 0001 0	001 0001 0001 00	01	22	444	1
0x666666	0110 0110 1	110 0110 0110 01	10	CC	1999	2
0xBBBBBB	1011 1011 0	011 1011 1011 10	11	177	2EEE	3

4.5 Consider a 32-bit microprocessor that has an on-chip 16-Kbyte four-way set-associative cache. Assume that the cache has a line size of four 32-bit words. Draw a block diagram of this cache showing its organization and how the different address fields are used to determine a cache hit/miss. Where in the cache is the word from memory location ABCDE8F8 mapped?

Answer:

32-bit microprocessor ➔ bits in an address = 32bits

Type of cache = set-associative

Number of lines in a set = 4 lines/set ➔ (four-way set-associative)

Size of line = four 32-bit words = 4 *32 bit = 4 * 4 * 8 bits = 16 bytes

Total size of Cache = 16 Kbytes

Number of bits in the address of one byte = 32 bits

Number of lines in cache = $\frac{\text{Total size of Cache}}{\text{Size of line}} = \frac{16 * 2^{10}}{16}$ = 1024 lines

Number of sets in cache = $\frac{\text{Number of lines in Cache}}{\text{Number of lines in a set}} = \frac{1024}{4}$ = 256 set

Number of bits for set field = log_2(Number of sets in cache) = $log_2(256)$ = 8 bits
Number of bits in offset (byte) field = log_2(Number of bytes in a line) = $log_2(16)$ = 4 bits
Number of bits in tag field = Number of bits in the address – (Number of bits for set field + Number of bits in offset (word) field)
= 32 – (8 + 4) = 20 bits

	Tag	Set	Offset (byte)
Address(32 bits total)	20 bits	8 bits	4 bits

	Tag	Set	Offset (byte)
	20 bits	8 bits	4 bits
Address(**ABCDE8F8**)	1010 1011 1100 1101 1110 1000 1111 1000		
	1010 1011 1100 1101 1110	1000 1111	1000

4.6 Given the following specifications for an external cache memory: four-way set associative; line size of two 16-bit words; able to accommodate a total of 4K 32-bit words from main memory; used with a 16-bit processor that issues 24-bit addresses. Design the cache structure with all pertinent information and show how it interprets the processor's addresses.

Answer:
Type of cache = set-associative
Number of lines in a set = 4 lines/set ➔ (four-way set-associative)
Size of line = 2 *16bit = 2 * 2 * 8bit = 4 byte
Total size of Cache = 4 K * 32 bit = 4k * 4 * 8bit=16384 bytes
Number of bits in the address = 24 bit
Number of lines in cache = $\frac{\text{Total size of Cache}}{\text{Size of line}} = \frac{16384}{4}$ = 4096 lines
Number of sets in cache = $\frac{\text{Number of lines in Cache}}{\text{Number of lines in a set}} = \frac{4096}{4}$ = 1024 set
Number of bits for set field = log_2(Number of sets in cache) = $log_2(1024)$ = 10 bits
Number of bits in offset (byte) field = log_2(Number of bytes in a line) = $log_2(4)$ = 2 bits
Number of bits in tag field = Number of bits in the address – (Number of bits for set field + Number of bits in offset (word) field)
= 24 – (10 + 2) = 12 bits

	Tag	Set	Offset (byte)
Address(24 bits total)	12 bits	10 bits	2 bits

4.7 The Intel 80486 has an on-chip, unified cache. It contains 8 Kbytes and has a four-way set-associative organization and a block length of four 32-bit words. The cache is organized into 128 sets. There is a single "line valid bit" and three bits, B0, B1, and B2 (the "LRU" bits), per line. On a cache miss, the 80486 reads a 16-byte line from main memory in a bus memory read burst. Draw a simplified diagram of the cache and show how the different fields of the address are interpreted.

Answer:
Type of cache = set-associative
Number of bits in address = 32 bits ➔ **Intel 80486 , not given**
Number of lines in a set = 4 lines/set ➔ (four-way set-associative)
Size of line = 4 * 32 bit = 4 * 4 * 8bits= 16 bytes
Total size of Cache = 8 Kbytes
Number of sets = 128 sets
Number of bits for set field = log_2(Number of sets in cache) = $log_2(128)$ = 7 bits
Number of bits in offset (byte) field = log_2(Number of bytes in a line) = $log_2(16)$ = 4 bits
Number of bits in tag field = Number of bits in the address – (Number of bits for set field + Number of bits in offset (word) field)
= 32 – (7 + 4) = 21 bits

	Tag	Set	Offset (byte)
Address(26 bits total)	21 bits	7 bits	4 bits

Each set contains 4 Lines + 3 LRU bits

Each Line contains 16 bytes + a valid bit + 21 bit tag

4.8 Consider a machine with a byte addressable main memory of 216 bytes and block size of 8 bytes. Assume that a direct mapped cache consisting of 32 lines is used with this machine.

a. How is a 16-bit memory address divided into tag, line number, and byte number?

b. Into what line would bytes with each of the following addresses be stored?

0001 0001 0001 1011

1100 0011 0011 0100

1101 0000 0001 1101

1010 1010 1010 1010

c. Suppose the byte with address 0001 1010 0001 1010 is stored in the cache. What are the addresses of the other bytes stored along with it?

d. How many total bytes of memory can be stored in the cache?

e. Why is also the tag stored in the cache?

Answer:

Total size of main memory = 216 bytes

Block size = 8 bytes

Type of cache = direct

Number of lines in cache= 32 lines

A –

Number of bits in address = 16 bit

Number of bits for line field = log_2(Number of lines in cache) = $log_2(32)$ = 5 bits

Number of bits in offset (byte) field = log_2(Number of bytes in a line) = $log_2(8)$ = 3 bits

Number of bits in tag field = Number of bits in the address – (Number of bits for set field + Number of bits in offset (word) field)

= 16 – (5 + 3) = 8 bits

	Tag	Line	Offset (byte)
Address(16 bits total)	8 bits	5 bits	3 bits

B –

Address	Tag (8)	Line (5)	Offset (3)	Line(in hex)
0001 0001 0001 1011	0001 0001	0001 1	011	3
1100 0011 0011 0100	1100 0011	0011 0	100	6
1101 0000 0001 1101	1101 0000	0001 1	101	3
1010 1010 1010 1010	1010 1010	1010 1	010	2

C – 0001 1010 0001 1010 ➔ line 3 with offsets from 000 to 111

All addresses ➔

0001 1010 0001 1000

0001 1010 0001 1001

0001 1010 0001 1010

0001 1010 0001 1011

0001 1010 0001 1100

0001 1010 0001 1101

0001 1010 0001 1110

0001 1010 0001 1111

} All come from memory together and are put into a single line of cache

D – Size of cache = Number of lines in cache * Size of one line = 32 * 8 = 256 bytes

E- Because two items with two different memory addresses can be stored in the same place in the cache. The tag is used to distinguish between them.

4.10 A set-associative cache has a block size of four 16-bit words and a set size of 2. The cache can accommodate a total of 4096 words. The main memory size that is cacheable is 64K * 32 bits. Design the cache structure and show how the processor's addresses are interpreted.

Answer:

Type of cache = set-associative

Number of lines in a set = 2 lines/set

Size of line = 4 *16bit = 4 * 2 * 8bit = 8 byte

Total size of Cache = 4096 * 16bit = 4096 * 2 * 8bit=8192 bytes

Total size of Main memory = 64 k * 32 bit = 64 * 2^{10} * 4 * 8bit=262144 bytes

Number of bits in the address = $log_2(\text{size of Main memory in bytes}) = log_2(262144) = 18$ bit

Number of lines in cache = $\frac{\text{Total size of Cache}}{\text{Size of line}} = \frac{8192}{8} = 1024$ lines

Number of sets in cache = $\frac{\text{Number of lines in Cache}}{\text{Number of lines in a set}} = \frac{1024}{2} = 512$ set

Number of bits for set field = $log_2(\text{Number of sets in cache}) = log_2(512) = 9$ bits

Number of bits in offset (byte) field = $log_2(\text{Number of bytes in a line}) = log_2(8) = 3$ bits

Number of bits in tag field = Number of bits in the address – (Number of bits for set field + Number of bits in offset (word) field)

= 18 – (9 + 3) = 6 bits

	Tag	Set	Offset (byte)
Address(18 bits total)	18 bits	9 bits	3 bits

4.11 Consider a memory system that uses a 32-bit address to address at the byte level, plus a cache that uses a 64-byte line size.

a. Assume a direct mapped cache with a tag field in the address of 20 bits. Show the address format and determine the following parameters: number of addressable units, number of blocks in main memory, number of lines in cache, size of tag.

b. Assume an associative cache. Show the address format and determine the following parameters: number of addressable units, number of blocks in main memory, number of lines in cache, size of tag.

c. Assume a four-way set-associative cache with a tag field in the address of 9 bits. Show the address format and determine the following parameters: number of addressable units, number of blocks in main memory, number of lines in set, number of sets in cache, number of lines in cache, size of tag.

Answer:

Number of bits in the address = 32 bit

Size of line = 64 byte

A –

Type of cache = direct

Number of bits in tag field= 20 bit

Number of bits in offset (byte) field = $log_2(\text{Number of bytes in a line}) = log_2(64) = 6$ bits

Number of bits for line field = Number of bits in the address – (Number of bits in offset (byte) field + Number of bits in tag field)

= 32 – (6+20) = 6 bits

	Tag	Line	Offset (byte)
Address(32 bits total)	20 bits	6 bits	6 bits

B –

Type of cache = Associative

Number of bits in offset (byte) field = $log_2(\text{Number of bytes in a line}) = log_2(64)$ = 6 bits

Number of bits in tag field = Number of bits in the address –Number of bits in offset (byte) field

= 32 – 6 = 26 bits

	Tag	Offset (byte)
Address(32 bits total)	26 bits	6 bits

C –

Type of cache = set-associative

Number of bits in tag field= 9 bit

Number of bits in offset (byte) field = $log_2(\text{Number of bytes in a line}) = log_2(64)$ = 6 bits

Number of bits for set field = Number of bits in the address – (Number of bits in offset (byte) field + Number of bits in tag field)

= 32 – (6+9) = 17 bits

	Tag	Set	Offset (byte)
Address(32 bits total)	9 bits	17 bits	6 bits

4.12 Consider a computer with the following characteristics: total of 1Mbyte of main memory; word size of 1 byte; block size of 16 bytes; and cache size of 64 Kbytes.

a. For the main memory addresses of F0010, 01234, and CABBE, give the corresponding tag, cache line address, and word offsets for a direct-mapped cache.

b. Give any two main memory addresses with different tags that map to the same cache slot for a direct-mapped cache.

c. For the main memory addresses of F0010 and CABBE, give the corresponding tag and offset values for a fully-associative cache.

d. For the main memory addresses of F0010 and CABBE, give the corresponding tag, cache set, and offset values for a two-way set-associative cache.

Answer:

Total size of main memory = 1 Mbyte

Word size = 1byte

Block size = 16 bytes

Total size of cache = 64 Kbytes

Number of bits in address = $log_2(\text{Number of bytes in main memory}) = log_2(1 * 2^{20})$ = 20 bit

Number of lines in cache = $\frac{\text{Total size of Cache}}{\text{Size of line}} = \frac{64 * 2^{10}}{16}$ = 4096 lines

Number of bits for Offset field= $log_2(\text{Number of bytes in a line}) = log_2(16)$ = 4 bit

A –

Type of cache = direct

Number of bits for Line field= $log_2(\text{Number of bytes in a line}) = log_2(4096)$ = 12 bit

Number of bits for Tag field = Number of bits in the address – (Number of bits in offset (byte) field + Number of bits in Line field)

= 20 – (12 + 4) = 4 bits

	Tag	Line	Offset (byte)
Address(20 bits total)	4 bits	12 bits	4 bits

Address	Tag (4)	Line (12)	Offset (4)
0x F0010	1111	0000 0000 0001	0000
0x01234	0000	0001 0010 0011	0100
0x CABBE	1100	1010 1011 1011	1110

B –

For 0x F0010: 0x00010, 0x10010, 0x20010, 0x30010, 0x40010 , ... etc

For 0x 01234: 0x11234, 0x21234, 0x31234, 0x41234, 0x51234, ... etc

For 0x CABBE: 0x0ABBE, 0x1ABBE, 0x2ABBE, 0x3ABBE, 0x4ABBE , ... etc

C –

Type of cache = Associative

Number of bits for Tag field = Number of bits in the address –Number of bits in offset (byte) field

= 20 – (4) = 16 bits

	Tag	Offset (byte)
Address(20 bits total)	16 bits	4 bits

Address	Tag (16)	Offset (4)
0x F0010	1111 0000 0000 0001	0000
0x01234	0000 0001 0010 0011	0100
0x CABBE	1100 1010 1011 1011	1110

C –

Type of cache = set-Associative

Number of lines in a set = 2 lines/set

Number of sets in cache = $\frac{\text{Number of lines in Cache}}{\text{Number of lines in a set}} = \frac{4096}{2}$ = 2048 set

Number of bits for set field = $log_2(\text{Number of sets in cache}) = log_2(2048) = 11$ bits

Number of bits for Tag field = Number of bits in the address – (Number of bits in offset (byte) field + Number of bits in set field)

= 20 – (11 + 4) = 5 bits

	Tag	Line	Offset (byte)
Address(20 bits total)	5 bits	11 bits	4 bits

Address	Tag (5)	Line (11)	Offset (4)
0x F0010	1111 0	000 0000 0001	0000
0x01234	0000 0	001 0010 0011	0100
0x CABBE	1100 1	010 1011 1011	1110