8.2 Solution

Each column address strobe causes $8 \times 4 = 32$ bytes to be transferred.
(*a*) Latency = 5 clock cycles or 12.5 ns
Total time = 5 + 8 = 13 clock cycles, or 32.5 ns.
(*b*) A second column strobe is needed to transfer the second burst of 32 bytes. Therefore:
Latency = 5 clock cycles or 12.5 ns
Total time = 5 + 8 + 2 + 8 = 23 clock cycles, or 57.5 ns. 或者26个clock cycle也算对。

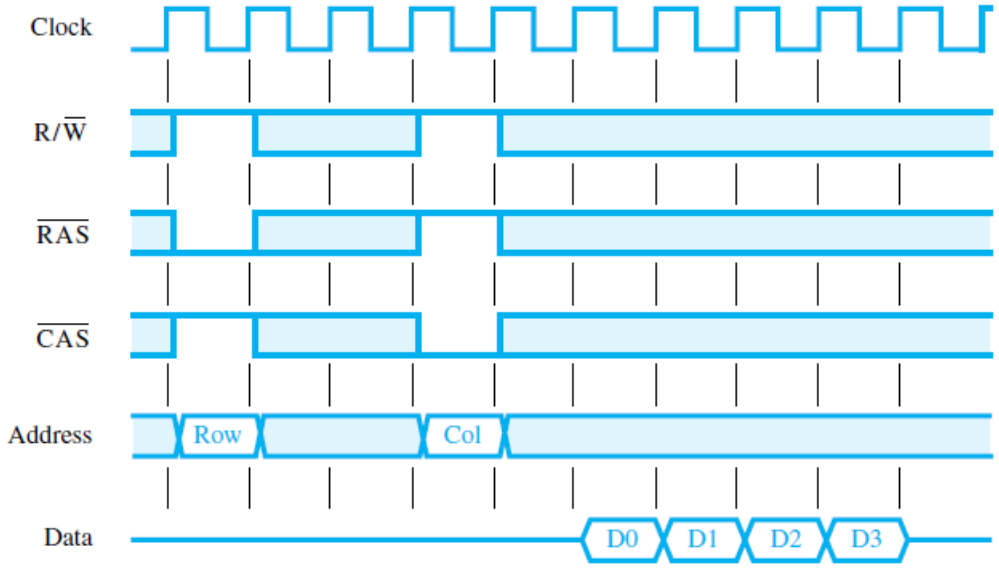


Figure 8.9 A burst read of length 4 in an SDRAM.

8.3 Solution

A 16M module can be structured as 16 rows, each containing eight $1M \times 4$ chips. A 24-bit address is required. Address lines A19-0 should be connected to all chips. Address lines A23-20 should be connected to a 4-bit decoder to select one of the 16 rows.

有的同学会画出类似图8.10的图，请注意地址线和数据线以及片选画的是否正确。

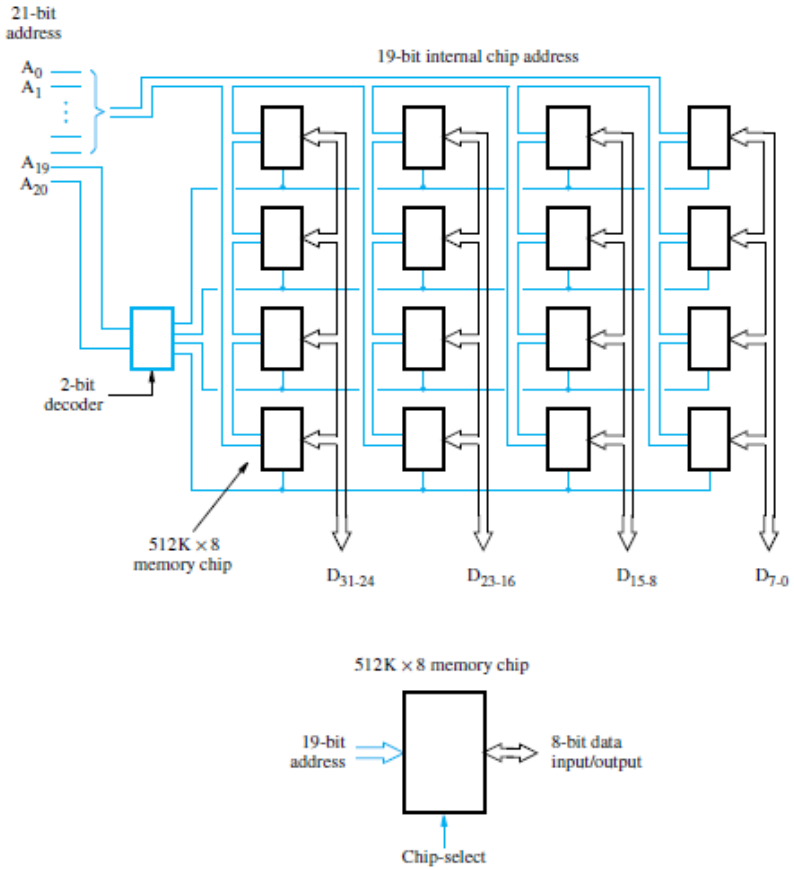


Figure 8.10 Organization of a 2M × 32 memory module using 512K × 8 static memory chips.

8.8 Solution

Each block contains 128 bytes, thus requiring a 7-bit Word field.

There are 16 sets, requiring a 4-bit Set field.

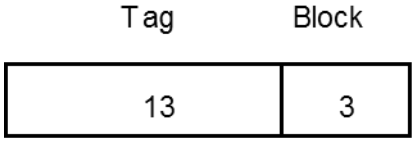
The remaining 21 bits of the address constitute the tag field.

8.10 Solution

For the first loop, the contents of the cache are as indicated in Figures 8.21 through 8.23. For the second loop, they are as follows.

	Memory Address	Contents
7A00	0111101000000000	A(0,0)
7A04	0111101000000100	A(0,1)
7A08	0111101000001000	A(0,2)
7A0C	0111101000001100	A(0,3)
7A10	0111101000010000	A(0,4)
7A14	0111101000010100	A(0,5)
7A18	0111101000011000	A(0,6)
7A1C	0111101000011100	A(0,7)
7A20	0111101000100000	A(0,8)
7A24	0111101000100100	A(0,9)

8.10 Solution (a) direct-mapped cache



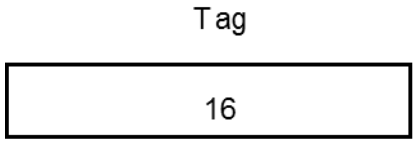
Main memory address

Block position	Contents of data cache after pass:					
	$j = 9$	$i = 1$	$i = 3$	$i = 5$	$i = 7$	$i = 9$
0	A(0,8)	A(0,0)	A(0,2)	A(0,4)	A(0,6)	A(0,8)
1						
2						
3						
4	A(0,9)	A(0,1)	A(0,3)	A(0,5)	A(0,7)	A(0,9)
5						
6						
7						

Block position	Contents of data cache after pass:								
	$j = 1$	$j = 3$	$j = 5$	$j = 7$	$j = 9$	$i = 6$	$i = 4$	$i = 2$	$i = 0$
0	A(0,0)	A(0,2)	A(0,4)	A(0,6)	A(0,8)	A(0,6)	A(0,4)	A(0,2)	A(0,0)
1									
2									
3									
4	A(0,1)	A(0,3)	A(0,5)	A(0,7)	A(0,9)	A(0,7)	A(0,5)	A(0,3)	A(0,1)
5									
6									
7									

Figure 8.21 Contents of a direct-mapped data cache.

8.10 Solution (b) associative-mapped cache



Block position	Contents of data cache after pass:				
	$j = 7$	$j = 8$	$j = 9$	$i = 1$	$i = 0$
0	A(0,0)	A(0,8)	A(0,8)	A(0,8)	A(0,0)
1	A(0,1)	A(0,1)	A(0,9)	A(0,1)	A(0,1)
2	A(0,2)	A(0,2)	A(0,2)	A(0,2)	A(0,2)
3	A(0,3)	A(0,3)	A(0,3)	A(0,3)	A(0,3)
4	A(0,4)	A(0,4)	A(0,4)	A(0,4)	A(0,4)
5	A(0,5)	A(0,5)	A(0,5)	A(0,5)	A(0,5)
6	A(0,6)	A(0,6)	A(0,6)	A(0,6)	A(0,6)
7	A(0,7)	A(0,7)	A(0,7)	A(0,7)	A(0,7)

Block position	Contents of data cache after pass:			
	$j = 9$	$i = 0$	$i = 5$	$i = 9$
0	A(0,8)	A(0,8)	A(0,8)	A(0,6)
1	A(0,9)	A(0,9)	A(0,9)	A(0,7)
2	A(0,2)	A(0,0)	A(0,0)	A(0,8)
3	A(0,3)	A(0,3)	A(0,1)	A(0,9)
4	A(0,4)	A(0,4)	A(0,2)	A(0,2)
5	A(0,5)	A(0,5)	A(0,3)	A(0,3)
6	A(0,6)	A(0,6)	A(0,4)	A(0,4)
7	A(0,7)	A(0,7)	A(0,5)	A(0,5)

Figure 8.22 Contents of an associative-mapped data cache.

8.10 Solution (c) set-associative-mapped cache

	Block position	Contents of data cache after pass:			
		$j = 9$	$i = 3$	$i = 7$	$i = 9$
Set 0	0	A(0,8)	A(0,2)	A(0,6)	A(0,6)
	1	A(0,9)	A(0,3)	A(0,7)	A(0,7)
	2	A(0,6)	A(0,0)	A(0,4)	A(0,8)
	3	A(0,7)	A(0,1)	A(0,5)	A(0,9)
Set 1	0				
	1				
	2				
	3				

	Contents of data cache after pass:					
	$j = 3$	$j = 7$	$j = 9$	$i = 4$	$i = 2$	$i = 0$
Set 0	A(0,0)	A(0,4)	A(0,8)	A(0,4)	A(0,4)	A(0,0)
	A(0,1)	A(0,5)	A(0,9)	A(0,5)	A(0,5)	A(0,1)
	A(0,2)	A(0,6)	A(0,6)	A(0,6)	A(0,2)	A(0,2)
	A(0,3)	A(0,7)	A(0,7)	A(0,7)	A(0,3)	A(0,3)
Set 1						

Figure 8.23 Contents of a set-associative-mapped data cache.

8.10 Conclusion:

In all 3 cases, all elements are overwritten before they are used in the second loop. This suggests that the LRU algorithm may not lead to good performance if used with arrays that do not fit into the cache.
Performance can be improved by introducing some randomness in the replacement algorithm.

8.11 Solution

Block position	Contents of data cache after:			
	Pass 1	Pass 2	Pass 3	Pass 4
0	[200]	[200]	[200]	[200]
1	[204]	[204]	[204]	[204]
2	[208]	[208]	[208]	[208]
3	[24C]	[24C]	[24C]	[24C]
4	[2F0]	[2F0]	[2F0]	[2F0]
5	[2F4]	[2F4]	[2F4]	[2F4]
6	[218]	[218]	[218]	[218]
7	[21C]	[21C]	[21C]	[21C]

Hit rate = 33/48 = 0.69

8.11 Solution (b) Associative-mapped cache

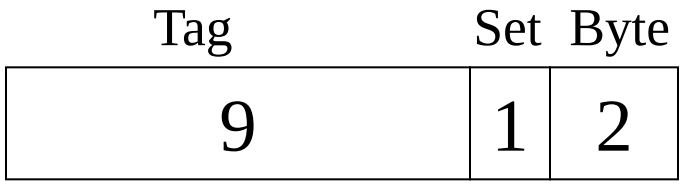
Tag	Byte
10	2

Main memory address

Block position	Contents of data cache after:			
	Pass 1	Pass 2	Pass 3	Pass 4
0	[200]	[200]	[200]	[200]
1	[204]	[204]	[204]	[204]
2	[24C]	[21C]	[218]	[2F0]
3	[20C]	[24C]	[21C]	[218]
4	[2F4]	[2F4]	[2F4]	[2F4]
5	[2F0]	[20C]	[24C]	[21C]
6	[218]	[2F0]	[20C]	[24C]
7	[21C]	[218]	[2F0]	[20C]

Hit rate = 21/48 = 0.44

8.11 Solution (c) Set-associative-mapped cache



Main memory address

	Block position	Contents of data cache after:			
		Pass 1	Pass 2	Pass 3	Pass 4
Set 0	0	[200]	[200]	[200]	[200]
	1	[208]	[208]	[208]	[208]
	2	[2F0]	[2F0]	[2F0]	[2F0]
	3	[218]	[218]	[218]	[218]
Set 1	0	[204]	[204]	[204]	[204]
	1	[24C]	[21C]	[24C]	[21C]
	2	[2F4]	[2F4]	[2F4]	[2F4]
	3	[21C]	[24C]	[21C]	[24C]

Hit rate = 30/48 = 0.63

8.13 Solution

Larger size

- fewer misses if most of the data in the block are actually used
- wasteful if much of the data are not used before the cache block is ejected from the cache
- Larger miss penalty

Smaller size

- more misses
- smaller miss penalty

8.22 Solution

(a) The maximum number of bytes that can be stored on this disk is $24 \times 14000 \times 400 \times 512 = 68.8 \times 10^9$ bytes.

(b) The data transfer rate is $(400 \times 512 \times 7200)/60 = 24.58 \times 10^6$ bytes/s.

Chapter8补充题

1.If a byte-addressable machine with 32-bit words stores the hex value 98765432 at address 0x00, indicate how this value would be stored on a little-endian machine and on a big-endian machine.

Solution:

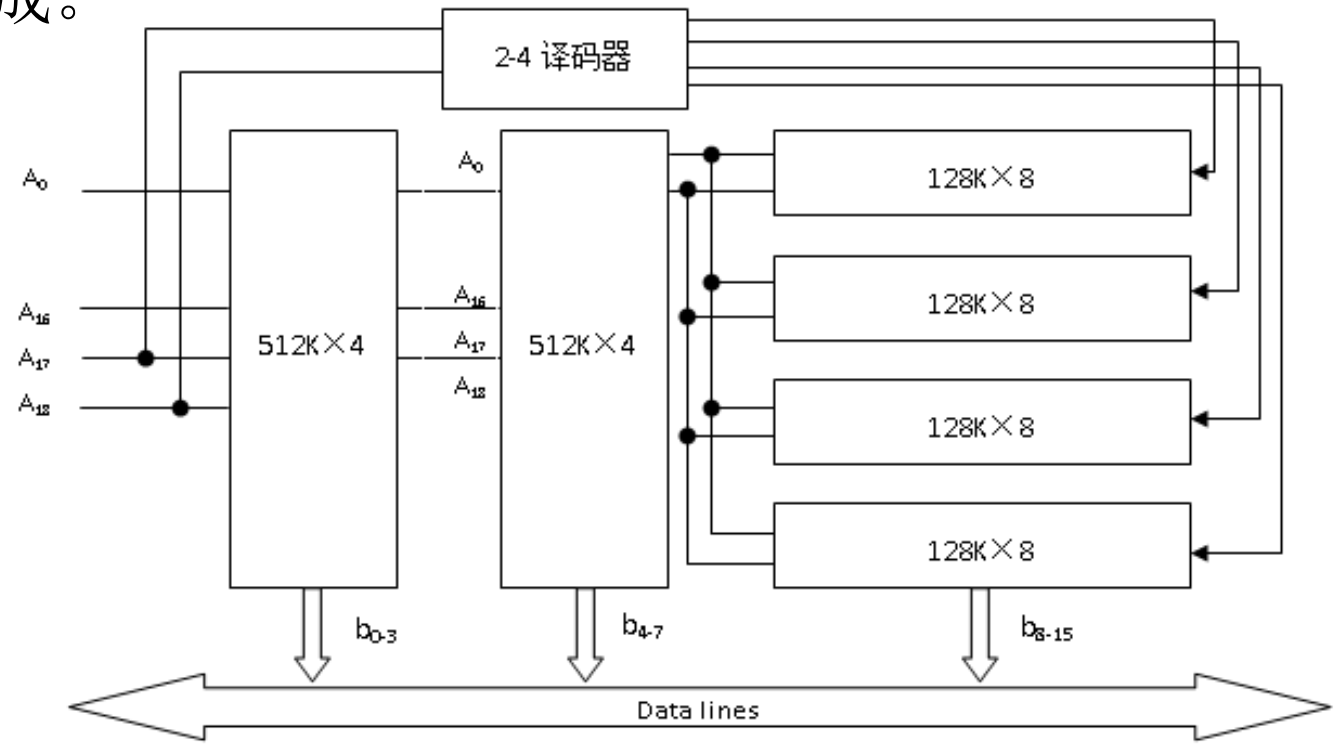
The value 98765432 in little and big endian machines would be stored as shown below

Byte Address	00 01 02 03
Little Endian	32 54 76 98
Big Endian	98 76 54 32

大容量存储器部分补充题

2. Assume that there are two types of static memory chips: 128K×8 bit (total 4 chips) and 512K×4 bit (total 2 chips). Please use these memory chips to implement a 512K×16 bit memory. Draw the figure of the memory organization.

Solution: 512K×16的存储器需要2片512K×4 和4片128K×8的芯片按下图组成。



Cache映射部分补充题

3. A computer system has a cache organized in set associative manner, with 4 blocks per set and 64 words per block. The cache consists of 32 blocks and the main memory consists of 1024 blocks. The main memory is word-addressable.

(1)How many bits are there in main memory address?

Solution:

Main memory 1024 blocks, need 10 bits to represent block address.
64 words per block, need 6 bits to represent word address.

So main memory address needs total 10+6=16 bits.

(2)Calculate the number of bits in each of the TAG, SET, and WORD fields of the main memory address format.

Solution:

64 words per block, so WORD field 6bits.

cache set 32/4=8, so SET field 3bits.

TAG field 16-6-3=7bits.

Cache映射部分补充题

3. A computer system has a cache organized in set associative manner, with 4 blocks per set and 64 words per block. The cache consists of 32 blocks and the main memory consists of 1024 blocks. The main memory is word-addressable.

(3)Assuming that the cache is initially empty. The processor fetches 3072 words sequentially from main memory locations 0, 1, …, 3071. Assume that the LRU algorithm is used for block replacement. Compute the hit rate of this cache.

Solution:主存单元0，1，2，…，3071，按64字为一块共48块，块号为0-47。从主存单元依次读出3072个字，相当于依次读出48个块。每次读块的第一个字是不命中的。由于每个块读完以后不再读了，所以命中率是（3072-48）/3072=98.4%

补充题

4. Consider the memory system with the following specifications:

Byte-addressable.

Virtual address space: 4G bytes.

Main memory size: 16M bytes.

Cache size: 512K bytes.

Page size: 64K bytes.

Block size: 32 bytes

Mapping Strategies:

Main Memory to Cache: 8-way set associative;

Hard Disk to Main Memory: fully associative.

Virtual address is first translated to physical address. Then, it accesses the cache memory using the physical address.

补充题（续）

4.

(1) How many sets are there in the cache memory?

(2) How long is the tag field of the cache?

(3) Given a virtual address 0547AF33 (hexadecimal), its corresponding virtual page is stored in physical page 3B (hexadecimal).

① What is its physical address under such mapping?

② Which set can this address be possibly found in the cache?

③ Which byte does this address point to out of the 32 bytes in a block?

Solution:

(1) 512K/(32×4)=2048

There are 2048 sets in the cache memory.

(2) Because the main memory is byte-addressable and the main memory size: 16M bytes, so the main memory address is 24-bit long.

From (1) there are 2048 sets in the cache memory, so the set field of the main memory address is 11-bit long. And because the block size: 32 bytes, so the byte field of the main memory address is 5-bit long. Finally, the tag field of the cache is 24－11－5 =8-bit long.

(3)

①There are 4G/64K=2^{16} virtual pages ,so the virtual page number is 16-bit long. Because the virtual address is 32-bit long, so the offset field of it is 32-16=16-bit long.

The virtual address 0547AF33 can be divided into two parts, the highest 16-bit virtual page number and the lowest 16-bit offset. So the offset field of this virtual address is AF33. Concatenate it with the physical page number. So the physical address under such mapping is 3BAF33.

②3BAF33=(0011 1011 1010 1111 0011 0011)$_2$

So this address can be found in 10101111001 or 1401 set in the cache.

③3BAF33=(0011 1011 1010 1111 0011 0011)$_2$

The byte number is 10011 or 19.