Exercise 1

Problem 1

Assume there is a **direct-mapped** cache with **4 sets**, **1 line** per set, **2 bytes** per block, and **4-bit addresses**. The following table enumerates the entire address space and partitions the bits.

	Address bits		
Address (decimal)	Tag bits ($t = 1$)	Index bits ($s = 2$)	Offset bits ($b = 1$)
0	0	00	0
1	0	00	1
2	0	01	0
3	0	01	1
4	0	10	0
5	0	10	1
6	0	11	0
7	0	11	1
8	1	00	0
9	1	00	1
10	1	01	0
11	1	01	1
12	1	10	0
13	1	10	1
14	1	11	0
15	1	11	1

Initially, the cache is empty (i.e., each valid bit is zero):

Set	Valid	Tag	block[0]	block[1]
0	0			
1	0			
2	0			
3	0			

Fill the cache with proper value when the CPU performs a sequence of reads: (Use m[i] to identify the value of address i in memory)

1. Read one byte at address 0.

Set	Valid	Tag	block[0]	block[1]
0	1	0	m[0]	m[1]
1	0			
2	0			
3	0			

2. Read one byte at address 1.

Set	Valid	Tag	block[0]	block[1]
0	1	0	m[0]	m[1]
1	0			
2	0			
3	0			

3. Read one byte at address 11.

Set	Valid	Tag	block[0]	block[1]
0	1	0	m[0]	m[1]
1	1	1	m[10]	m[11]
2	0			
3	0			

4. Read one byte at address 12.

Set	Valid	Tag	block[0]	block[1]
0	1	0	m[0]	m[1]
1	1	1	m[10]	m[11]
2	1	1	m[12]	m[13]
3	0			

5. Read one byte at address 4.

Set	Valid	Tag	block[0]	block[1]
0	1	0	m[0]	m[1]
1	1	1	m[10]	m[11]
2	1	0	m[4]	m[5]
3	0			

Problem 2

Jack has a **32-bit** machine with a **2-way** set associative cache. There are **8 sets**. Each block is **4 bytes**. The following table shows the content of the data cache at time T. **`ByteX`** is the byte value stored at offset **x**.

Set	Tag	Valid	Byte0	Byte1	Byte2	Byte3	Tag	Valid	Byte0	Byte1	Byte2	Byte3
0	0x5df	1	0x11	0x22	0xfe	0x43	0x5d2	1	0xca	0xdb	0xed	0x00
1	0x7cf	1	0xab	0xcd	0xef	0xff	0x34e	1	0xdf	0x11	0x22	0x33
2	0x233	0	0x23	0x32	0x23	0x33	0x34e	1	0xfd	0x44	0x55	0x66
3	0x435	1	0xde	0xad	0xbe	0xef	0x34e	1	0xdf	0x11	0x22	0x33
4	--	0	--	--	--	--	--	0	--	--	--	--
5	0x701	1	0xff	0xff	0xcc	0xcc	0x435	1	0xad	0x18	0x24	0x19
6	0x881	1	0xde	0xed	0xbe	0xef	0x781	0	0x23	0x32	0xff	0xdd
7	--	0	--	--	--	--	--	0	--	--	--	--

1. How would a **32-bit** physical memory address be split into tag/set-index /block-offset fields in this machine?

   tag _27_ bits, set-index __3_ bits, and block-offset __2_ bits

2. What is the size of this cache in bytes? 64 bytes

3. Assume the cache line replacement policy is **LRU**. A short program will read memory in the following sequences starting from time T. Each access will read **one byte**. Please fill the following blanks and compute the miss rate. If there is a cache miss, enter '--' for 'Byte Returned'.

Order	Address	Set	Hit/Miss	Byte Returned
1	0xbbe0	0	Hit	0x11
2	0x66a3	0	Miss	--
3	0xf039	6	Miss	--
4	0x69c6	1	Hit	0x22
5	0xba41	0	Miss	--

Miss rate: ___0.6__