Given that we have the cache below; using 6-bit address "direct mapped" architecture, and "write-back" policy for updating the cache, answer the following questions:

Cache Index	Dirty Bit	Valid Bit	Tag	DATA
00	0	0	00	4 Bytes
01	0	0	01	4 Bytes
10	0	0	10	4 Bytes
11	0	0	00	4 Bytes

1) If the CPU tries to load the word at the following address: 010100

1a) The cache needs to (for some reason) go to 2nd level cache:
True or False

1b) Fill the table below after executing the 'load' instruction:

Cache Index	Dirty Bit	Valid Bit	Tag	DATA
00	0	0	00	4 Bytes
01	0	1	01	4 Bytes
10	0	0	10	4 Bytes
11	0	0	00	4 Bytes

2) If the CPU tries to store a word at the following address: 010100

2a) The cache needs to (for some reason) go to 2nd level cache:
True or False

2b) Fill the table below executing the 'store' instruction:

Cache Index	Dirty Bit	Valid Bit	Tag	DATA
00	0	0	00	4 Bytes
01	1	1	01	4 Bytes
10	0	0	10	4 Bytes
11	0	0	00	4 Bytes

3) If the CPU tries to load the word at the following address: 110100

3a) The cache needs to (for some reason) go to 2nd level cache:
True or False

3b) Fill the table below after executing the 'load' instruction:

Cache Index	Dirty Bit	Valid Bit	Tag	DATA
00	0	0	00	4 Bytes
01	1	1	~~01~~ 11	4 Bytes
10	0	0	10	4 Bytes
11	0	0	00	4 Bytes

Assuming that our address bus is 16 bits. Fill the table below

Tag Size	Block Size	Block Offset Size	Number of Blocks/Rows	Number of Comparators	Set Index Size	Cache Size	Cache Architecture
	16		32				Fully Associative
		2	128				Fully Associative
		3			4		Direct Mapped
	128					512 Bytes	Direct Mapped
8	32						Direct Mapped

First row:

Tag Size	Block Size	Block Offset Size	Number of Blocks/Rows	Number of Comparators	Set Index Size	Cache Size	Cache Architecture
12	16	4	32	32	0	512 Bytes	Fully Associative

- Since the block size is 16, we know that the block offset is 4. $2^4 = 16$
- The number of comparators in "Fully Associative" is the same as the number of rows: 32 in this example.
- Since we're not decoding the address to determine the exact block/row in "Fully Associative", that means the "set index" size is 0
- Cache size is Block size * Number of blocks: 16 * 32 = 512 Bytes
- Tag size is address length minus "block offset" size, minus "set index" size, which would be 16 - 4 - 0 = 12

Second row:

Tag Size	Block Size	Block Offset Size	Number of Blocks/Rows	Number of Comparators	Set Index Size	Cache Size	Cache Architecture
14	4	2	128	128	0	512 Bytes	Fully Associative

- Since the block offset size is 2, we know that the block size is 4. $2^2 = 4$
- The number of comparators in "Fully Associative" is the same as the number of rows: 128 in this example.
- Since we're not decoding the address to determine the exact block/row in "Fully Associative", that means the "set index" size is 0
- Cache size is Block size * Number of blocks: 4 * 128 = 512 Bytes
- Tag size is address length minus "block offset" size, minus "set index" size, which would be 16 - 2 - 0 = 14

Third row:

Tag Size	Block Size	Block Offset Size	Number of Blocks/Rows	Number of Comparators	Set Index Size	Cache Size	Cache Architecture
9	8	3	16	1	4	128 Bytes	Direct Mapped

- Since the block offset size is 3, we know that the block size is 8. $2^3 = 8$
- The number of comparators in "Direct Mapped" is 1
- Since we are decoding the "set index" to determine the exact block/row in "Direct Mapped", that means the number of blocks/rows is $2^{4\text{(set index size)}} = 16$
- Cache size is Block size * Number of blocks: 8 * 16 = 128 Bytes
- Tag size is address length minus "block offset" size, minus "set index" size, which would be 16 - 3 - 4 = 9

Fourth row:

Tag Size	Block Size	Block Offset Size	Number of Blocks/Rows	Number of Comparators	Set Index Size	Cache Size	Cache Architecture
7	128	7	4	1	2	512 Bytes	Direct Mapped

- Since the block size 128, we know that the block offset size is 6. $2^7 = 128$
- The number of comparators in "Direct Mapped" is 1
- Since we have the cache size and the block size, we can get the number of blocks/rows. Dividing the cache size (512) by the block size (128) gives us 4
- Since we are decoding the "set index" to determine the exact block/row in "Direct Mapped", that means the size of the set index is 2. $2^2 = 4$
- Tag size is address length minus "block offset" size, minus "set index" size, which would be 16 - 7 - 2 = 7

Fifth row:

Tag Size	Block Size	Block Offset Size	Number of Blocks/Rows	Number of Comparators	Set Index Size	Cache Size	Cache Architecture
8	32	5	8	1	3	256 Bytes	Direct Mapped

- Since the block size 32, we know that the block offset size is 5. $2^5 = 32$
- The number of comparators in "Direct Mapped" is 1
- Since tag size is address length minus "block offset" size, minus "set index" size, that means our set index is 3. Set index = 16 - 5 - 8
- Since we are decoding the "set index" to determine the exact block/row in "Direct Mapped", that means the number of blocks/rows is 8. $2^3 = 8$
- Cache size is Block size * Number of blocks: 32 * 8 = 256 Bytes