CS 31 Homework 6: Caching – due Apr 5

In this assignment, you will be tracing memory accesses for two different cache organizations. The systems have the following architecture characteristics:

- Both have 8-bit memory addresses
- Both have a 2-byte cache block size
- Cache A is 4-line Direct Mapped.
- Cache B is is 2-way Set Associative, with a total of 4 sets.

Question 1

How many bytes of data can be stored in each of the following? Do not include metadata like the tag, dirty bit, valid bit, or LRU bit.

(a) Cache A

(b) Cache B

Question 2

For each of the two cache organizations, show how the bits of the address are divided into the `tag`, `index/set`, and `byte offset`.

(a) Cache A

```
1 0 1 0 1 0 1 0
```

(b) Cache B

```
1 0 1 0 1 0 1 0
```

Question 3

On the accompanying diagram for Cache Organization A (direct mapped), show the results of the following memory operations (R: read, W: write). Within each box, time should progress downward, so the first address loaded appears at the top and subsequent changes are written below. To the right of the table, label each change with number of the operation that caused it. Annotate each operation below with *hit* or *miss* to indicate whether the data was found in the cache. Don't forget to update the dirty and valid bits! **Note:** Do not worry about what is read or what is written: you are only showing how the meta-data in the cache changes when performing these operations.

1. `R 0 0 0 1 1 0 1 0`
2. `W 0 0 0 1 1 0 1 1`
3. `R 1 1 1 1 1 0 0 0`
4. `R 1 1 1 1 1 0 1 0`
5. `R 0 1 1 0 1 0 0 0`
6. `W 0 0 0 0 1 0 0 1`
7. `R 0 0 0 0 0 0 0 0`
8. `W 0 0 0 1 1 0 1 0`

Question 4

On the accompanying diagram for Cache Organization B (2-way set associative), show the results of the following memory operations. Within each box, time should progress downward, so the first address loaded appears at the top and subsequent changes are written below. To the right of the table, label each change with number of the operation that caused it. Annotate each operation below with *hit* or *miss* to indicate whether the data was found in the cache. Don't forget to update the dirty, valid, and LRU bits! Note that the valid bit for each entry in the cache begins at 0.

1. `R 0 0 0 1 1 0 1 0`
2. `W 0 0 0 1 1 0 1 1`
3. `R 1 1 1 1 1 0 0 0`
4. `R 1 1 1 1 1 0 1 0`
5. `R 0 1 1 0 1 0 0 0`
6. `W 0 0 0 0 1 0 0 1`
7. `R 0 0 0 0 0 0 0 0`
8. `W 0 0 0 1 1 0 1 0`

Cache A

index	dirty	valid	tag
0		0	
1		0	
2		0	
3		0	

Cache B

set	LRU	D	V	tag	D	V	tag
0	1		0			0	
1	0		0			0	
2	0		0			0	
3	1		0			0	