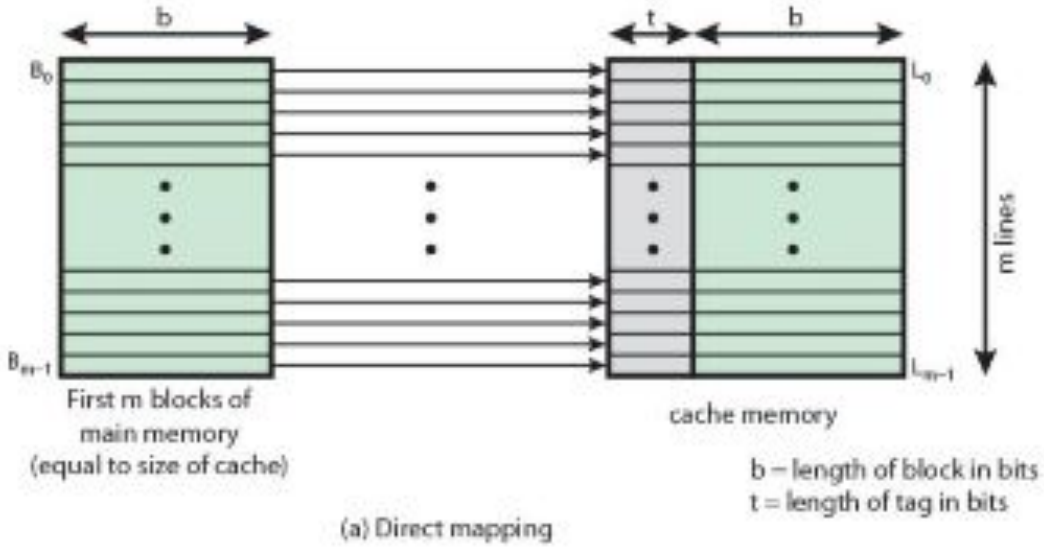
Post Lab 10

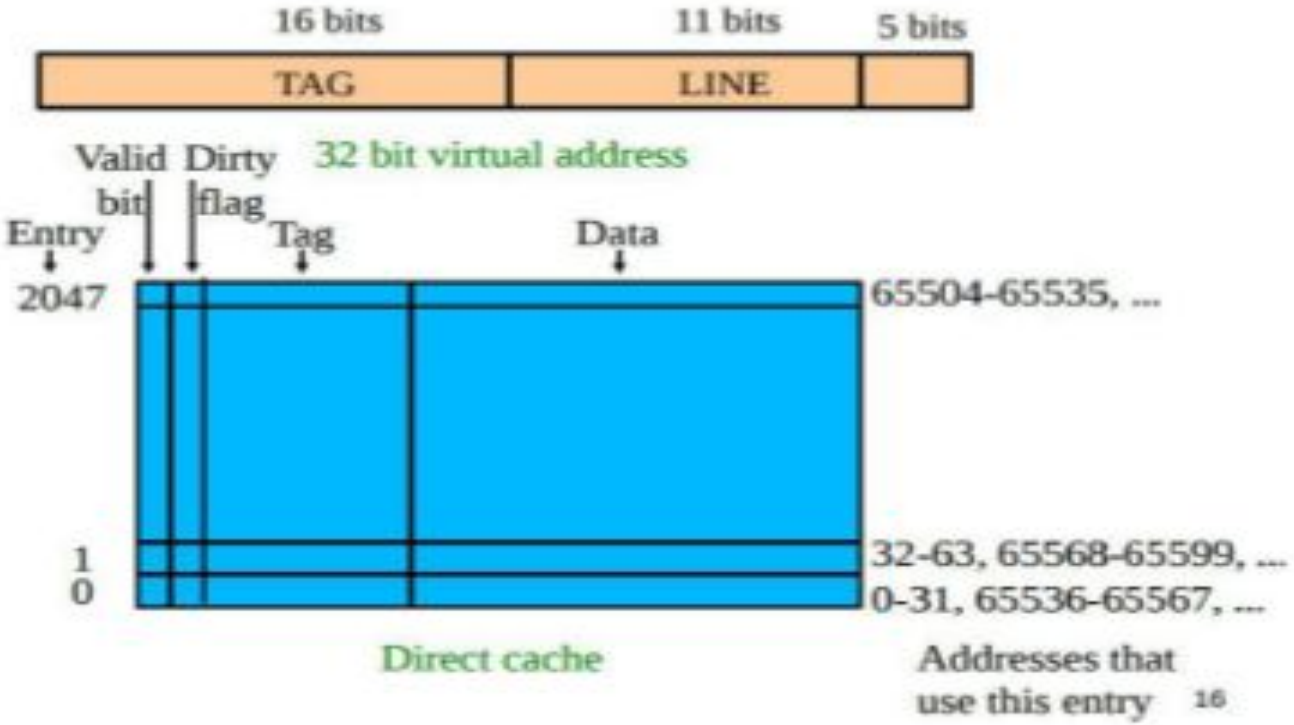
NAME - Rajendra Singh
Roll no. - 111601017

(1) Direct mapped cache: The following code fragment simulates a direct-mapped cache with 8 lines of 1 word

Compile and execute the direct-mapped cache simulator given above. Put appropriate comments for the code sections and report the final number of hits, accesses and hit rate output by the cod



(a) Direct mapping



The detail operation of **direct mapping** technique is as follows: The main memory address is divided into three fields. The field size depends on the memory capacity and the block size of **cache**. In this example, the lower 5 bits of address is used to identify a word within a block.

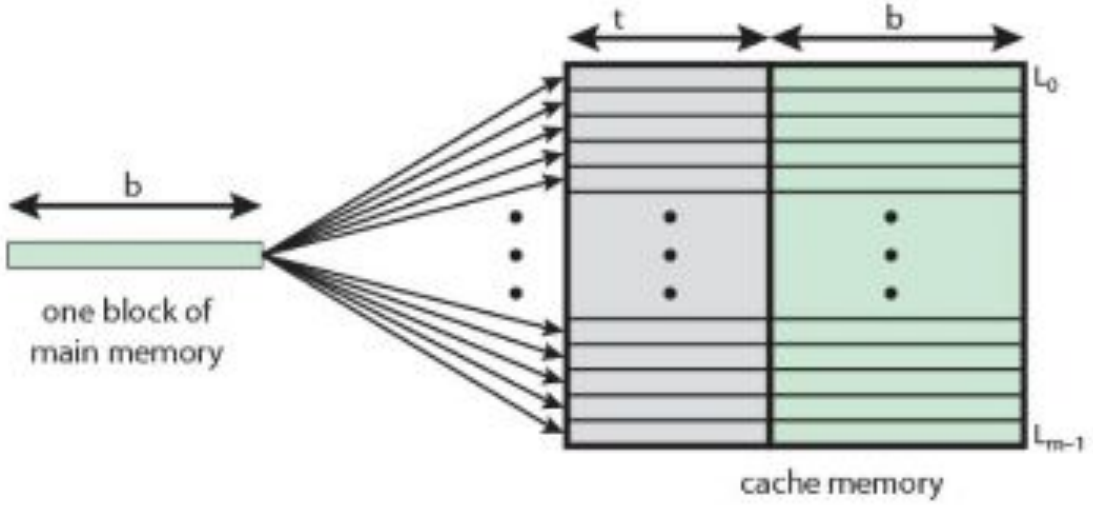
```c
#include <stdio.h>
int tag[8];
int main( )
{
    int addr;
    int i, j, t;
    int hits, accesses;
    FILE *fp;
    fp = fopen("trace.txt", "r");
    hits = 0;
    accesses = 0;
    while (fscanf(fp, "%x", &addr) > 0) {
        /* simulate a direct-mapped cache with 8 words */
        accesses += 1;
        printf("%3d: 0x%08x ", accesses, addr);
        i = (addr >> 2) & 7;
        t = addr | 0x1f;
        if (tag[i] == t) {
            hits += 1;
            printf("Hit%d ", i);
        } else {
            /* allocate entry */
```

```
15          printf("%3d: 0x%08x ", accesses, addr);
16          i = (addr >> 2) & 7;
17          t = addr | 0x1f;
18          if (tag[i] == t) {
19              hits += 1;
20              printf("Hit%d ", i);
21          } else {
22              /* allocate entry */
23              printf("Miss ");
24              tag[i] = t;
25
26          }
27          for (i = 0; i < 8; i++)
28              printf("0x%08x ", tag[i]);
29          printf("\n");
30      }
31
32  printf("Hits=%d,Accesses=%d,Hitratio=%f\n",hits,accesses,((float)hits)/accesses);
33      fclose(fp);
34  }
35
36
```

```
 67: 0x80000014 Hit5 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000003f
 68: 0x80000018 Hit6 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000003f
 69: 0x8000000c Hit3 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000003f
 70: 0x00000054 Miss 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x0000005f 0x8000001f 0x0000003f
 71: 0x80000010 Hit4 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x0000005f 0x8000001f 0x0000003f
 72: 0x80000014 Miss 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000003f
 73: 0x80000018 Hit6 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000003f
 74: 0x8000000c Hit3 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000003f
 75: 0x00000058 Miss 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x0000005f 0x0000003f
 76: 0x80000010 Hit4 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x0000005f 0x0000003f
 77: 0x80000014 Hit5 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x0000005f 0x0000003f
 78: 0x80000018 Miss 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000003f
 79: 0x8000000c Hit3 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000003f
 80: 0x0000005c Miss 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 81: 0x80000010 Hit4 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 82: 0x80000014 Hit5 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 83: 0x80000018 Hit6 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 84: 0x8000000c Hit3 0x0000005f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 85: 0x00000060 Miss 0x0000007f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 86: 0x80000010 Hit4 0x0000007f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 87: 0x80000014 Hit5 0x0000007f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 88: 0x80000018 Hit6 0x0000007f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 89: 0x8000000c Hit3 0x0000007f 0x0000005f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 90: 0x00000064 Miss 0x0000007f 0x0000007f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 91: 0x80000010 Hit4 0x0000007f 0x0000007f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 92: 0x80000014 Hit5 0x0000007f 0x0000007f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 93: 0x80000018 Hit6 0x0000007f 0x0000007f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 94: 0x8000000c Hit3 0x0000007f 0x0000007f 0x0000005f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 95: 0x00000068 Miss 0x0000007f 0x0000007f 0x0000007f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 96: 0x80000010 Hit4 0x0000007f 0x0000007f 0x0000007f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 97: 0x80000014 Hit5 0x0000007f 0x0000007f 0x0000007f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 98: 0x80000018 Hit6 0x0000007f 0x0000007f 0x0000007f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
 99: 0x8000000c Hit3 0x0000007f 0x0000007f 0x0000007f 0x8000001f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
100: 0x0000006c Miss 0x0000007f 0x0000007f 0x0000007f 0x0000007f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
101: 0x80000010 Hit4 0x0000007f 0x0000007f 0x0000007f 0x0000007f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
102: 0x80000014 Hit5 0x0000007f 0x0000007f 0x0000007f 0x0000007f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
103: 0x80000018 Hit6 0x0000007f 0x0000007f 0x0000007f 0x0000007f 0x8000001f 0x8000001f 0x8000001f 0x0000005f
Hits=68,Accesses=103,Hitratio=0.660194
rajendra@Singh:~/Desktop/co2610/lab10new$
```

(2) Fully Associative Cache: The following code fragment simulates a fully associative cache with 8 lines each of 1 word, with least recently used (LRU) cache replacement algorithm. The ‘mru’ in the code fragment represents most recently used.

Compile and execute the fully associative cache simulator given above. Put appropriate comments for the code sections and report the final number of hits, accesses and hit rate output by the code.



fully associative cache

A cache where data from any address can be stored in any cache location. The whole address must be used as the tag. All tags must be compared simultaneously (associatively) with the requested address and if one matches then its associated data is accessed. This requires an associative memory to hold the tags which makes this form of cache more expensive. It does however solve the problem of contention for cache locations since a block need only be flushed when the whole cache is full and then the block to flush can be selected in a more efficient way.

```
#include <stdio.h>
    int tag[8];
    int mru[8] = {7,6,5,4,3,2,1,0};
    void mruUpdate(int index) {
          int i;
          for (i = 0; i < 8; i++)
if (mru[i] == index)
break;
          while (i > 0) {
mru[i] = mru[i-1];
i--;    }
          mru[0] = index;     }
     int main( ){
 int addr;
 int i, j, t;
 int hits, accesses;
 FILE *fp;
 fp = fopen("trace.txt", "r");
 hits = 0;
 accesses = 0;
 while (fscanf(fp, "%x", &addr) > 0) {
        accesses += 1;
```

```
        accesses += 1;
        printf("%3d: 0x%08x ", accesses, addr);
        for (i = 0; i < 8; i++) {
            if (tag[i] == addr) {
                hits += 1;
                printf("Hit%d ", i);
                mruUpdate(i);
                break;
            }
        }
        if (i == 8) {
            printf("Miss ");
            i = mru[7];
            tag[i] = addr;
            mruUpdate(i);
        }
        for (i = 0; i < 8; i++)
            printf("0x%08x ", tag[i]);
        for (i = 0; i < 8; i++)
            printf("%d ", mru[i]);
        printf("\n");
```

```
                hits += 1;
                printf("Hit%d ", i);
                mruUpdate(i);
                break;
            }
        }
        if (i == 8) {
            printf("Miss ");
            i = mru[7];
            tag[i] = addr;
            mruUpdate(i);
        }
        for (i = 0; i < 8; i++)
            printf("0x%08x ", tag[i]);
        for (i = 0; i < 8; i++)
            printf("%d ", mru[i]);
        printf("\n");
    }

printf("Hits=%d,Accesses=%d,Hitratio=%f\n",hits,accesses, ((float)hits)/accesses);
    fclose(fp); }

```

```
 67: 0x80000014 Hit6 0x00000044 0x00000048 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 6 5 4 3 7 2 1 0
 68: 0x80000018 Hit7 0x00000044 0x00000048 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 7 6 5 4 3 2 1 0
 69: 0x8000000c Hit3 0x00000044 0x00000048 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 3 7 6 5 4 2 1 0
 70: 0x00000054 Miss 0x00000054 0x00000048 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 0 3 7 6 5 4 2 1
 71: 0x80000010 Hit5 0x00000054 0x00000048 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 5 0 3 7 6 4 2 1
 72: 0x80000014 Hit6 0x00000054 0x00000048 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 6 5 0 3 7 4 2 1
 73: 0x80000018 Hit7 0x00000054 0x00000048 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 7 6 5 0 3 4 2 1
 74: 0x8000000c Hit3 0x00000054 0x00000048 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 3 7 6 5 0 4 2 1
 75: 0x00000058 Miss 0x00000054 0x00000058 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 1 3 7 6 5 0 4 2
 76: 0x80000010 Hit5 0x00000054 0x00000058 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 5 1 3 7 6 0 4 2
 77: 0x80000014 Hit6 0x00000054 0x00000058 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 6 5 1 3 7 0 4 2
 78: 0x80000018 Hit7 0x00000054 0x00000058 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 7 6 5 1 3 0 4 2
 79: 0x8000000c Hit3 0x00000054 0x00000058 0x0000004c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 3 7 6 5 1 0 4 2
 80: 0x0000005c Miss 0x00000054 0x00000058 0x0000005c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 2 3 7 6 5 1 0 4
 81: 0x80000010 Hit5 0x00000054 0x00000058 0x0000005c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 5 2 3 7 6 1 0 4
 82: 0x80000014 Hit6 0x00000054 0x00000058 0x0000005c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 6 5 2 3 7 1 0 4
 83: 0x80000018 Hit7 0x00000054 0x00000058 0x0000005c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 7 6 5 2 3 1 0 4
 84: 0x8000000c Hit3 0x00000054 0x00000058 0x0000005c 0x8000000c 0x00000050 0x80000010 0x80000014 0x80000018 3 7 6 5 2 1 0 4
 85: 0x00000060 Miss 0x00000054 0x00000058 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 4 3 7 6 5 2 1 0
 86: 0x80000010 Hit5 0x00000054 0x00000058 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 5 4 3 7 6 2 1 0
 87: 0x80000014 Hit6 0x00000054 0x00000058 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 6 5 4 3 7 2 1 0
 88: 0x80000018 Hit7 0x00000054 0x00000058 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 7 6 5 4 3 2 1 0
 89: 0x8000000c Hit3 0x00000054 0x00000058 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 3 7 6 5 4 2 1 0
 90: 0x00000064 Miss 0x00000064 0x00000058 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 0 3 7 6 5 4 2 1
 91: 0x80000010 Hit5 0x00000064 0x00000058 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 5 0 3 7 6 4 2 1
 92: 0x80000014 Hit6 0x00000064 0x00000058 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 6 5 0 3 7 4 2 1
 93: 0x80000018 Hit7 0x00000064 0x00000058 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 7 6 5 0 3 4 2 1
 94: 0x8000000c Hit3 0x00000064 0x00000058 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 3 7 6 5 0 4 2 1
 95: 0x00000068 Miss 0x00000064 0x00000068 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 1 3 7 6 5 0 4 2
 96: 0x80000010 Hit5 0x00000064 0x00000068 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 5 1 3 7 6 0 4 2
 97: 0x80000014 Hit6 0x00000064 0x00000068 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 6 5 1 3 7 0 4 2
 98: 0x80000018 Hit7 0x00000064 0x00000068 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 7 6 5 1 3 0 4 2
 99: 0x8000000c Hit3 0x00000064 0x00000068 0x0000005c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 3 7 6 5 1 0 4 2
100: 0x0000006c Miss 0x00000064 0x00000068 0x0000006c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 2 3 7 6 5 1 0 4
101: 0x80000010 Hit5 0x00000064 0x00000068 0x0000006c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 5 2 3 7 6 1 0 4
102: 0x80000014 Hit6 0x00000064 0x00000068 0x0000006c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 6 5 2 3 7 1 0 4
103: 0x80000018 Hit7 0x00000064 0x00000068 0x0000006c 0x8000000c 0x00000060 0x80000010 0x80000014 0x80000018 7 6 5 2 3 1 0 4
Hits=76,Accesses=103,Hitratio=0.737864
rajendra@Singh:~/Desktop/co2610/lab10new$
```

(3) Two Way Set Associative Cache: Use the following code fragment as a basis for implementing a 2-way set associative cache with LRU replacement

Where the 8 tags are stored as a 2 by 4 entry array, where the first array index selects between the lines in the 2-way set and the second index selects one of 4 lines. The second array, mru[], tracks the most recently used of the two lines in each set. Finish the supplied program for simulating a 2-way set associative cache with LRU replacement using the trace in the "trace.txt" file. Compile and execute it. Put appropriate comments for the code sections and report the final number of hits, accesses and hit rate output by the code. How will you approach for a 4 way set associative cache?

```c
#include<stdio.h>

int tag[2][4];
int mru[4] = {1,1,1,1};

void mruUpdate(int i,int j){

  mru[i] = j;

}


int main(){

int addr,i,j,t;
int hits, accesses;
FILE *fp;

fp = fopen("trace.txt","r");
hits = 0;
accesses = 0;

```

```
22
23    while(fscanf(fp,"%x",&addr) > 0){
24
25    accesses += 1;
26    printf("%3d: 0x%08x ", accesses, addr);
27
28    i = (addr >> 2) & 3; // get block number or column index
29
30    //printf("%d\n",i);
31
32      for(j=0;j<2;j++){
33
34        if(tag[j][i] == addr){
35
36          hits += 1;
37          printf("Hit%d ",i);
38          //mru Update
39          mru[i] = j;
40          break;
41        }
42      }
43
```

```
45      if (j == 2){
46
47
48        printf("Miss ");
49        int least_index;
50
51        if(mru[i] == 1)
52          least_index = 0;
53
54        else
55          least_index = 1;
56
57        tag[least_index][i] = addr;
58
59        // mru update
60        mru[i] = least_index;
61
62      }
63
64
65      for(int n=0;n<2;n++){
```

```
66          for(int m=0;m<4;m++){
67            printf("0x%08x ", tag[n][m]);
68
69          }
70        }
71
72        printf("  ");
73
74        for(int n=0;n<4;n++)
75          printf("%d ",mru[n]);
76
77        printf("\n");
78
79    }
80
81  printf("Hits = %d, Accesses = %d, Hit ratio = %f\n", hits, accesses, ((float)hits)/accesses);
82
83  fclose(fp);
84
85
86  }
87
```

```
 67: 0x80000014 Hit1 0x80000010 0x00000044 0x00000048 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 1 1 0
 68: 0x80000018 Hit2 0x80000010 0x00000044 0x00000048 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 1 1 0
 69: 0x8000000c Hit3 0x80000010 0x00000044 0x00000048 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 1 1 0
 70: 0x00000054 Miss 0x80000010 0x00000054 0x00000048 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 0 1 0
 71: 0x80000010 Hit0 0x80000010 0x00000054 0x00000048 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 0 1 0
 72: 0x80000014 Hit1 0x80000010 0x00000054 0x00000048 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 1 1 0
 73: 0x80000018 Hit2 0x80000010 0x00000054 0x00000048 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 1 1 0
 74: 0x8000000c Hit3 0x80000010 0x00000054 0x00000048 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 1 1 0
 75: 0x00000058 Miss 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 1 0 0
 76: 0x80000010 Hit0 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 1 0 0
 77: 0x80000014 Hit1 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 1 0 0
 78: 0x80000018 Hit2 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 1 1 0
 79: 0x8000000c Hit3 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000004c   0 1 1 0
 80: 0x0000005c Miss 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000005c   0 1 1 1
 81: 0x80000010 Hit0 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000005c   0 1 1 1
 82: 0x80000014 Hit1 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000005c   0 1 1 1
 83: 0x80000018 Hit2 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000005c   0 1 1 1
 84: 0x8000000c Hit3 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000050 0x80000014 0x80000018 0x0000005c   0 1 1 0
 85: 0x00000060 Miss 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   1 1 1 0
 86: 0x80000010 Hit0 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 1 0
 87: 0x80000014 Hit1 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 1 0
 88: 0x80000018 Hit2 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 1 0
 89: 0x8000000c Hit3 0x80000010 0x00000054 0x00000058 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 1 0
 90: 0x00000064 Miss 0x80000010 0x00000064 0x00000058 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 0 1 0
 91: 0x80000010 Hit0 0x80000010 0x00000064 0x00000058 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 0 1 0
 92: 0x80000014 Hit1 0x80000010 0x00000064 0x00000058 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 1 0
 93: 0x80000018 Hit2 0x80000010 0x00000064 0x00000058 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 1 0
 94: 0x8000000c Hit3 0x80000010 0x00000064 0x00000058 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 1 0
 95: 0x00000068 Miss 0x80000010 0x00000064 0x00000068 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 0 0
 96: 0x80000010 Hit0 0x80000010 0x00000064 0x00000068 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 0 0
 97: 0x80000014 Hit1 0x80000010 0x00000064 0x00000068 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 0 0
 98: 0x80000018 Hit2 0x80000010 0x00000064 0x00000068 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 1 0
 99: 0x8000000c Hit3 0x80000010 0x00000064 0x00000068 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000005c   0 1 1 0
100: 0x0000006c Miss 0x80000010 0x00000064 0x00000068 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000006c   0 1 1 1
101: 0x80000010 Hit0 0x80000010 0x00000064 0x00000068 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000006c   0 1 1 1
102: 0x80000014 Hit1 0x80000010 0x00000064 0x00000068 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000006c   0 1 1 1
103: 0x80000018 Hit2 0x80000010 0x00000064 0x00000068 0x8000000c 0x00000060 0x80000014 0x80000018 0x0000006c   0 1 1 1
Hits = 76, Accesses = 103, Hit ratio = 0.737864
rajendra@Singh:~/Desktop/co2610/lab10new$
```