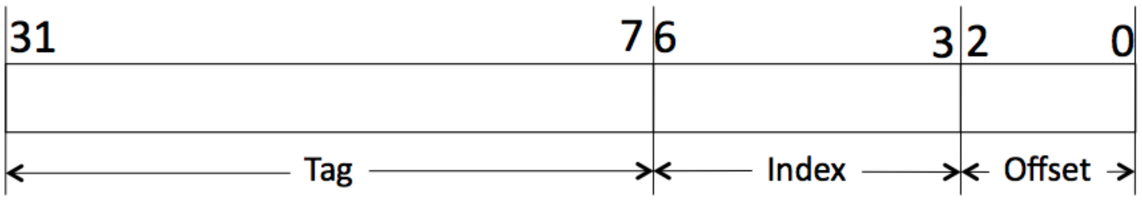
Question 1 (35 points): TinyProc Inc. is a new company in the market delivering tiny processors for the toy robot market. One of its processors, TP512, has an split data and instruction level 1 cache. The data cache is two-way set associative and the instruction cache directly mapped. Both caches use the following address mapping:



1. (**10 points**) What is the storage capacity, expressed in bytes, of each cache?

   In both caches, each block has $2^3 = 8$ bytes.

   The instruction cache (direct mapped) has $2^4 = 16$ entries, each with a single block, thus it can store $2^7 = 128$ bytes.

   The data cache (2-way set associative) also has $2^4 = 16$ entries, but each entry has two blocks (one for each set), thus it can store $2^8 = 256$ bytes.

2. (**10 points**) All TinyProc Inc processors use the MIPS instruction set. A *synthetic benchmark* is a program that does not do any useful computation, instead it is created to enable computer architects to analyze the performance of individual components of a design such as the branch predictor or the memory hierarchy. Assume that the following synthetic benchmark, which executes an infinite loop, is executed in TP512. Execution starts at address `0x8000 0000`.

```
0x8000 0000             add   $v0, $zero, $zero
0x8000 0004             lui   $s0, 0x4000
0x8000 0008  snippet1:  lw    $v0, 0($s0)
0x8000 000C             lw    $t2, 4($s0)
0x8000 0010             add   $v0, $v0, $t2
0x8000 0014             j     snippet2
0x8000 0018             add   $v0, $v0, $v0
                        ....
0x8000 0080  snippet2:  lw    $t2, 128($s0)
0x8000 0084             add   $v0, $v0, $t2
0x8000 0088             lw    $t2, 132($s0)
0x8000 008C             add   $v0, $v0, $t2
0x8000 0090             j     snippet1
0x8000 0094             jr    $ra
```

   In the tables below, indicate the memory addresses (expressed in hexadecimal), the tag, the index, and the offset for each access performed to the instruction and to the data cache by this synthetic benchmark. You must list the accesses into the tables in the order in which they occur in the program. Under the columns **First Iteration** and **Second Iteration** indicate if the access results in a hit or a miss in that iteration (list the outcome of accesses that occur before the first iteration of the loop in the **First Iteration** column and use "—" if the access does not occur in the iteration). For the tag you can use a combination of hexadecimal and binary notation, such as `0xAF78 87_01` where the digits before the "_" symbol are in hexadecimal and the ones after the"_" are in binary. There might be more lines in each of the tables than the number of accesses performed by the program in one iteration. Leave extra lines blank.

Instruction Cache:

Memory Address	**Tag**	**index**	**Offset**	**First Iteration**	**Second Iteration**
0x8000 0000	0x8000 00_0	000 0	000	Miss	—
0x8000 0004	0x8000 00_0	000 0	100	Hit	—
0x8000 0008	0x8000 00_0	000 1	000	Miss	Miss
0x8000 000C	0x8000 00_0	000 1	100	Hit	Hit
0x8000 0010	0x8000 00_0	001 0	000	Miss	Miss
0x8000 0014	0x8000 00_0	001 0	100	Hit	Hit
0x8000 0080	0x8000 00_1	000 0	000	Miss	Hit
0x8000 0084	0x8000 00_1	000 0	100	Hit	Hit
0x8000 0088	0x8000 00_1	000 1	000	Miss	Miss
0x8000 008C	0x8000 00_1	000 1	100	Hit	Hit
0x8000 0090	0x8000 00_1	001 0	000	Miss	Miss

Data Cache:

Memory Address	**Tag**	**index**	**Offset**	**First Iteration**	**Second Iteration**
0x4000 0000	0x4000 00_0	000 0	000	Miss	Hit
0x4000 0004	0x4000 00_0	000 0	100	Hit	Hit
0x4000 0080	0x4000 00_1	000 0	000	Miss	Hit
0x4000 0084	0x4000 00_1	000 0	100	Hit	Hit

3. (**10 points**) Assume that the synthetic benchmark's infinite loop is run for a long time and that the hit ratios of the instruction and data cache are measured during this run. What would be the measured ratios?

   - There are 9 access to the instruction cache within the loop and 5 of those are hits. Therefore, the hit ratio for the instruction cache is $\frac{5}{9}$ or 55%.
   - The hit ratio for that data cache is 100%.

4. (**5 points**) If the instruction cache was replaced with a 2-way set associative cache that uses the same address mapping shown above, what would be its new hit ratio for the synthetic program above? Explain your answer.

   The new hit ratio would be 100% because the infinite loop in the synthetic benchmark references at most two distinct blocks for each index value. A two-way set associative cache could contain both blocks and thus would hit every reference after the first iteration of the infinite loop.