(Exercise) Direct-map cache

1. Cache is a way to store and read data and a really high rate. We need it because it makes input and output speeds increase, and it is faster to access important memory when needed.
2. n/a
3. n/a
4. We need 4 bits for the offset; $log_2(blocksize) = offset$; no, because we need more information to determine the offset, tag, and index.
5. We need 8 bits for the index; $log_2(blocksize * 4) = index$
6. Yes, the equation for the total amount of cache is (# of blocks * size of one block)
7. We create a tag, this allows us to compare to the address that we want to access and see if it is the correct one, and if not we throw a miss.
8.
   a.
   b. 3 bits
   c. 16 blocks
   d. 5 bits
   e. 24 bits
   f.

Tag	Valid	Dirty	Blocks
14 bits	1 bit	1 bit	64 bit

9.
   g. 32 bits

(Exercise) N-way set associative cache

1.) The blocks are mapped to a fixed location, and you have to keep constantly go to the virtual memory to get the data you need. If you have two blocks that are mapped to the same point, this will create thrashing, its page faulting.

2.) N/A

3.) Index bits needed 8

$log_2(\frac{Block\ Size}{Associativity})$

4.)

a.) $2^{20} = 1048576$ bytes

b.) $Log_2 8 = 3$ bits

c.) $\frac{128}{8}$ = 16 blocks

d.) $\frac{Block\ Size}{Associativity} = \frac{16}{2}$ = 8 sets

e.) 3 bits

f.) 20 - (index + offset) = 14 bits

g.)

Tag	Valid	Dirty	DataBlocks
14 bits	1 bit	1 bit	64 bits

h.) 80 bits

(Assignment, individual) Cache in your computer

1.) There are 3 levels of cache, and L1 cache is separate for data and instructions
2.) L1 combined = 64K ; L2 = 256K ; L3 = 8192K
3.) 64 blocks
4.) Set associative; 3 sets
5.) There is no index bit; tag = 54; offset = 10