Homework 4

5.2

> A memory's addressability is 64 bits. What does that tell you about the size of the MAR and MDR?

MAR: not know

MDR: 64 bits

5.4

> Say we have a memory consisting of 256 locations, and each location contains 16 bits.
>
> *a.* How many bits are required for the address?
>
> *b.* If we use the PC-relative addressing mode, and want to allow control transfer between instructions 20 locations away, how many bits of a branch instruction are needed to specify the PC-relative offset?
>
> *c.* If a control instruction is in location 3, what is the PC-relative offset of address 10? Assume that the control transfer instructions work the same way as in the LC-3.

a. 8 bits

b. 6 bits

c. 6

5.9

> We would like to have an instruction that does nothing. Many ISAs actually have an opcode devoted to doing nothing. It is usually called NOP, for NO OPERATION. The instruction is fetched, decoded, and executed. The execution phase is to do nothing! Which of the following three instructions could be used for NOP and have the program still work correctly?
>
> *a.* 0001 001 001 1 00000

ADD R1,R1,#0
not NOP, it sets conditional codes.

> *b.* 0000 111 000000001

BRnzp #1
not NOP, jump to the next address of PC

> *c.* 0000 000 000000000

BR, never go in, do nothing.
It's NOP.

> What does the ADD instruction do that the others do not do?

sets conditional codes

5.15

register	content
R1	0011 0001 0010 0001
R2	0100 0101 0110 0110
R3	1010 1011 1100 1101
R4	1010 1011 1100 1101

5.16

> Which LC-3 addressing mode makes the most sense to use under the following conditions? (There may be more than one correct answer to each of these; therefore, justify your answers with some explanation.)
>
> *a.* You want to load one value from an address that is less than ±2^8 locations away.

PC-relative
Cause 2^8 can be represented by 9 bits, it's small and convenient.

> *b.* You want to load one value from an address that is more than 2^8 locations away.

Indirect
It's beyond 9 bits, so we have to use a Intermediate amount to go that far.

> *c.* You want to load an array of sequential addresses

Base+offset
Because PC-relative and indirect modes are all relative to the current address, while base-offset mode is about a fixed address.
Base-offset mode is more convenient to find an array which has a fixed start address. Because then we just need to add 1 to offset continuously instead of calculating the PC-offset with PC and goal-address changing.