Exercises:

Part I

1. Represent 305 with 10 bits
2. Represent -305 with 10 bits
3. Represent -197 with 9 bits
4. Represent 197 with 9 bits
5. Represent “0101010101” in hexadecimal
6. Represent “1100111101” in hexadecimal
7. Represent “11001100” as decimal (unsigned representation)
8. Represent “11001100” as decimal (signed representation)

Part II

1. Assume ***b*** is a variable of size 4 bytes and is stored in a byte addressable memory at address 0xa80. If the processor’s endianness is little-endian, and the processor writes the value 0xa155f0d3 in the variable ***b*** which bytes will be written to each memory address.

2. Assume ***x*** is a variable of type pointer that points to a single byte. Further, assume ***b*** is a variable of size 4 bytes and is stored at memory address 0xa80. Given that the processor uses big-endian, evaluate the new value of b after the following code is executed:

```
b=0x2F552;
x=0xa81;
b=b+*x;
```

3. Given a variable ***b*** which is assumed to have a value in the range [0..7], write the necessary statements in C to ensure that the bit at bit position ***b*** in another variable ***c*** is set to one.

4. Write a statement in C such that for a given variable ***b*** the bit at position 3 is set to 0, the bit at position 5 is set to 1, the bit at bit position 2 is inverted. Assume that the variable ***b*** is of size 1 byte.