```
############################################################
#           cs315 Week 5 - part 2
#
#   ->  Logical and arithmetic operations (or bitwise operations)
#
############################################################
```

Truth tables for logical operations:

A	B	A AND B	A OR B	A XOR B	NOT A
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

Note: if we are given *immediate* instruction then:
1) if we are given logical operation (AND, OR, XOR, NOT, ...), then sign extend it with zeros.
2) if we are given arithmetic operation (ADD), sign extend it based on its sign (which is the 16th bit)

Ex: if we are given this set of register values:
$t1 <-- 0xF123 1230 : 1111 0001 0010 0011 0001 0010 0011 0000
$t2 <-- 0x1FFF 3211 : 0001 1111 1111 1111 0011 0010 0001 0001

```
addi $t0, $t1, 0xFFF0

1111 1111 1111 1111 1111 111
1111 0001 0010 0011 0001 0010 0011 0000
1111 1111 1111 1111 1111 1111 1111 0000
---------------------------------------
1111 0001 0010 0011 0001 0010 0010 0000 --> $t0 = 0x7123 1220


addi $t0, $t1, 0x123E

                       1   1    11
1111 0001 0010 0011 0001 0010 0011 0000
0000 0000 0000 0000 0001 0010 0011 1110
---------------------------------------
1111 0001 0010 0011 0010 0100 0110 1110 --> $t0 = 0xF123 246E



andi $t0, $t1, 0x123E

1111 0001 0010 0011 0001 0010 0011 0000
0000 0000 0000 0000 0001 0010 0011 1110
---------------------------------------
0000 0000 0000 0000 0001 0010 0011 0000 --> $t0 = 0x0000 1230


ori $t0, $t1, 0x123E

1111 0001 0010 0011 0001 0010 0011 0000
0000 0000 0000 0000 0001 0010 0011 1110
---------------------------------------
1111 0001 0010 0011 0001 0010 0011 1110 --> $t0 = 0xF123 123E


xori $t0, $t1, 0x123E

1111 0001 0010 0011 0001 0010 0011 0000
0000 0000 0000 0000 0001 0010 0011 1110
---------------------------------------
1111 0001 0010 0011 0000 0000 0000 1110 --> $t0 = 0xF123 000E


-----------------------------------------------------------------------

add $t0, $t1, $t2

1111 1111 1111 111   11   1     1
1111 0001 0010 0011 0001 0010 0011 0000
0001 1111 1111 1111 0011 0010 0001 0001
---------------------------------------
0001 0001 0010 0010 0100 0100 0100 0001 --> $t0 = 0x1122 4441
```

```
xor $t0, $t1, $t2

1111 0001 0010 0011 0001 0010 0011 0000
0001 1111 1111 1111 0011 0010 0001 0001
---------------------------------------
1110 1110 1101 1100 0010 0000 0010 0001 --> $t0 = 0xEEDC 2021
```