COMP1036 Computer Fundamentals
Lab 3

Below is a list of all the chip interfaces in the Hack chip-set. If you need to use a chip-part, you can copy-paste the chip interface and proceed to fill in the missing data. This is a very useful list to have bookmarked or printed.

```
Add16(a= ,b= ,out= );
ALU(x= ,y= ,zx= ,nx= ,zy= ,ny= ,f= ,no= ,out= ,zr= ,ng= );
And16(a= ,b= ,out= );
And(a= ,b= ,out= );
ARegister(in= ,load= ,out= );
Bit(in= ,load= ,out= );
CPU(inM= ,instruction= ,reset= ,outM= ,writeM= ,addressM= ,pc= );
DFF(in= ,out= );
DMux4Way(in= ,sel= ,a= ,b= ,c= ,d= );
DMux8Way(in= ,sel= ,a= ,b= ,c= ,d= ,e= ,f= ,g= ,h= );
DMux(in= ,sel= ,a= ,b= );
DRegister(in= ,load= ,out= );
FullAdder(a= ,b= ,c= ,sum= ,carry= );
HalfAdder(a= ,b= ,sum= , carry= );
Inc16(in= ,out= );
Keyboard(out= );
Memory(in= ,load= ,address= ,out= );
Mux16(a= ,b= ,sel= ,out= );
Mux4Way16(a= ,b= ,c= ,d= ,sel= ,out= );
Mux8Way16(a= ,b= ,c= ,d= ,e= ,f= ,g= ,h= ,sel= ,out= );
Mux(a= ,b= ,sel= ,out= );
Nand(a= ,b= ,out= );
Not16(in= ,out= );
Not(in= ,out= );
Or16(a= ,b= ,out= );
Or8Way(in= ,out= );
Or(a= ,b= ,out= );
PC(in= ,load= ,inc= ,reset= ,out= );
RAM16K(in= ,load= ,address= ,out= );
RAM4K(in= ,load= ,address= ,out= );
RAM512(in= ,load= ,address= ,out= );
RAM64(in= ,load= ,address= ,out= );
RAM8(in= ,load= ,address= ,out= );
Register(in= ,load= ,out= );
ROM32K(address= ,out= );
```

```
Screen(in= ,load= ,address= ,out= );
Xor(a= ,b= ,out= );
```

1. Work out the representation for the following unsigned numbers by hand.

   (a) 45

   (b) 1026

   Solution

   ```
   remainder
   45 / 2 1
   22  / 2 0
   11  / 2 1
   5   / 2 1
   2   / 2 0
   1   / 2 1
   0

   The result is 101101

   remainder
   1026 / 2 0
   513  / 2 1
   256 / 2 0
   128 / 2 0
   64 / 2 0
   32 / 2 0
   16 / 2 0
   8 / 2 0
   4 / 2 0
   2 / 2 0
   1 / 2 1
   0

   The result is 10000000010
   ```

2. Work out the two's complement representations for the following signed numbers by hand(using 16-bits representation).

   (a) 26

   Solution

   ```
   0000000000011010
   ```

(b) -130

Solution

```
130(decimal) = 0000000010000010(binary)
Take bit wise inverse = 1111111101111101
Add 1 = 1111111101111110
```

3. Implement the following circuits:

(a) HalfAdder

```
Half adder. Computes sum, the least significant bit of a + b,
and carry, the most significant bit of a + b.
```

Solution

```
CHIP HalfAdder {
IN a, b;    // 1-bit inputs
OUT sum,    // Right bit of a + b
    carry;  // Left bit of a + b

PARTS:
Xor(a=a, b=b, out=sum);
And(a=a, b=b, out=carry);
}
```

(b) FullAdder

```
Full adder. Computes sum, the least significant bit of
a + b + c, and carry, the most significant bit of a + b + c.
```

Solution

```
CHIP FullAdder {
IN a, b, c;  // 1-bit inputs
OUT sum,     // Right bit of a + b + c
    carry;   // Left bit of a + b + c

PARTS:
HalfAdder(a=a, b=b, sum=w1, carry=c1);
HalfAdder(a=w1, b=c, sum=sum, carry=c2);
Xor(a=c1, b=c2, out=carry);
}
```

(c) Add16

```
Adds two 16-bit values.
The most-significant carry bit is ignored.
```

Solution

```
    CHIP Add16 {
    IN a[16], b[16];
    OUT out[16];

    PARTS:
    HalfAdder(a=a[0], b=b[0], sum=out[0], carry=carry1);
    FullAdder(a=a[1], b=b[1], c=carry1, sum=out[1], carry=carry2);
    FullAdder(a=a[2], b=b[2], c=carry2, sum=out[2], carry=carry3);
    FullAdder(a=a[3], b=b[3], c=carry3, sum=out[3], carry=carry4);
    FullAdder(a=a[4], b=b[4], c=carry4, sum=out[4], carry=carry5);
    FullAdder(a=a[5], b=b[5], c=carry5, sum=out[5], carry=carry6);
    FullAdder(a=a[6], b=b[6], c=carry6, sum=out[6], carry=carry7);
    FullAdder(a=a[7], b=b[7], c=carry7, sum=out[7], carry=carry8);
    FullAdder(a=a[8], b=b[8], c=carry8, sum=out[8], carry=carry9);
    FullAdder(a=a[9], b=b[9], c=carry9, sum=out[9], carry=carry10);
    FullAdder(a=a[10], b=b[10], c=carry10, sum=out[10], carry=carry11);
    FullAdder(a=a[11], b=b[11], c=carry11, sum=out[11], carry=carry12);
    FullAdder(a=a[12], b=b[12], c=carry12, sum=out[12], carry=carry13);
    FullAdder(a=a[13], b=b[13], c=carry13, sum=out[13], carry=carry14);
    FullAdder(a=a[14], b=b[14], c=carry14, sum=out[14], carry=carry15);
    FullAdder(a=a[15], b=b[15], c=carry15, sum=out[15], carry=carry16);
}
```