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. Initially, use the DFF chip as discussed in class. Examine what happens at the output on the tick and tock steps when different inputs are used. Then, using the inbuilt DFF chip, implement the following circuits:

   (a) 1-bit register

   Solution

```
CHIP Bit {
IN in, load;
OUT out;

PARTS:
Mux(a=t1, b=in, sel=load, out=w1);
DFF(in=w1, out=t1, out=out);
}
```

   (b) 16-bit register

```
If load[t-1]=1 then out[t] = in[t-1]
else out does not change (out[t] = out[t-1])
```

   Solution

```
CHIP Register {
IN in[16], load;
OUT out[16];

PARTS:
Bit(in=in[0], load=load, out=out[0]);
Bit(in=in[1], load=load, out=out[1]);
Bit(in=in[2], load=load, out=out[2]);
Bit(in=in[3], load=load, out=out[3]);
Bit(in=in[4], load=load, out=out[4]);
Bit(in=in[5], load=load, out=out[5]);
Bit(in=in[6], load=load, out=out[6]);
Bit(in=in[7], load=load, out=out[7]);
Bit(in=in[8], load=load, out=out[8]);
Bit(in=in[9], load=load, out=out[9]);
Bit(in=in[10], load=load, out=out[10]);
Bit(in=in[11], load=load, out=out[11]);
Bit(in=in[12], load=load, out=out[12]);
Bit(in=in[13], load=load, out=out[13]);
Bit(in=in[14], load=load, out=out[14]);
```

```
        Bit(in=in[15], load=load, out=out[15]);
    }
```

2. 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
```

3. 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
```