ICS Exercise

October 17, 2023

1 Bit Operations

Use **one formula** to implement the functions below. You are only allowed to use the given operations(**while/for/if** is not allowed). The integer is 32 bits.

1.1

Given an unsigned integer, swap all odd bits with even bits.
Example: swapAdj(23)=43.
Legal ops: & | << >>

```
unsigned int swapAdj(unsigned int x)
{
    return ((x & 0x55555555) << 1)
        | ((x & 0xaaaaaaaa) >> 1);
}
```

1.2

Find the rightmost different bit of x and y. Set the corresponding bit to 1 and others to 0 of the return value.
Example: diffRight(17, 34)=1. diffRight(80, 52)=4.
Legal ops: & $\sim$ ^ + -

```
int diffRight(int x, int y)
{
    return (x^y)&(~(x^y)+1);
}
```

2 Reverse bits

Write a function that reverses bits of a given 32 bits unsigned integer.

For example, given 00000010100101000001111010011100(2), your function should return 00111001011110000010100101000000(2).

```
uint32_t reverseBits(uint32_t n)
{
    int result = 0;
```

```
    for (int i = 0; i < 32; i++)
    {
        result <<= 1;
        result |= n & 1;
        n >>= 1;
    }
    return result;
}
```

3 Function Naming

1) Below are two poorly named functions written by ICS students, please give them proper function names according to their functionalities.

```
int f1 (int x, int y)
{
    return ((x&y) + ((x^y)>>1));
}
```

```
int f2 (int x, int y)
{
    int z = x - y;
    int k = (z >> 31) & 1;
    int m = x - k * z;
    return m;
}
```

Name of f1: average

Name of f2: max

2) Do the functions above provide their intended functionalities for all valid parameters? Why? Please explain with concrete examples.

f1 calculates the average of parameter x and y correctly. For example, f1 (-1,-1)=-1, f1(2147483647,1)=1073741824.

f2 does not return maximum one of parameter x and y when (x-y) is overflow. For example, f2(2147483647,-1)=-1.