Csed 211 fall 2021. Lab1

20200445 박은하

bitAnd함수

-x&y를 ~와 |만을 이용해 구현하였다.

X와 Y를 bit element로 이루어진 set로 생각하면 x&y는 x와 y집합의 교집합으로 생각할 수 있고 따라서 ~X와 ~Y의 합집합의 여집합으로 생각할 수 있었다.

따라서 x&y를 ~(~x|~y)로 구현하였다.

addOK함수

x와 y를 더했을 때 overflow가 발생하는지를 판단할 수 있는 함수이다.

X와y의 부호가 다르면 x^y의 msb는 1이 된다.

X와 Y의 부호가 다르면 x^y의 msb는 0이 된다.

따라서 x^y를 31만큼 right shift하면 부호가 다를 시 11111~이되고 부호가 같을 시 00000~이 된다. Shift한 값에 &1을 하면 부호가 다를 시 값이 1이 되어 1의 값이 리턴 된다. 이는 x와 y의 부호가 다를 경우 오버플로우가 일어나지 않는다는 법칙과 일치한다.

부호가 같을 시 x+y한 값의 msb와 x와 y의 msb가 다를 경우 overflow가 발생한다.

따라서 x와 x+y의 부호가 다르면 x^(x+y)의 msb는 1이된다.

X와 x+y의 부호가 같으면 x^(x+y)의 msb는 0이된다.

따라서 x^(x+y)를 31만큼 right shift하면 부호가 다를시 1111~이되고 부호가 같을 시 0000이된다. 부호가 다를 경우 over flow가 발생해야 하기 때문에 shift한 값에 ~를 취해주면 부호가 다른 경우 000~이되고 같으면 1111~이된다. 같을 시 1값을 리턴해야하므로 &1을 해주고 이를 아까 ((x^y)>>31)&1해주었던 값 뒤에 ‘|’ operator를 통해 연산하고 최종 값을 리턴한다.

IsNegative함수

Int는 signed이기 때문에 msb가 1인 경우 음수이다.따라서 x를 31만큼 right shift시키면 양수인경우 0000~이나오고 음수인 경우 1111~이 나온다. 음수일 경우 1을 리턴해야 하기에 shift한 값에 &1을 해준다.

logicalShift함수

x를 y만큼 logical right shift한 결과를 리턴하는 함수이다.

음수일 경우 y비트만큼 삭제되고 삭제된 만큼 좌측에 1이 생성되는데 이를 0으로 바꿔주기 위해 y개수만큼 앞자리에 0의 비트 값을 지닌 수를 만들어주었다.

((1<<31)>>y)<<1하면 msb부터 y개의 비트가 비트값 1을 가지게 되고 그 뒤의 비트들은 0의 값을 지닌다. 따라서 이에~를 취하면 y개수만큼 앞자리에 0의 비트 값을 지닌 수를 만들 수 있다.

이를 x>>y에 & operation을 통해 연산하면 logical right shift의 결과를 얻을 수 있다.

Bitcount함수

word에서 1의 개수를 return하는 함수이다. 1의 개수를 세주는 detecter역할을 하는 a라는 지역 변수를 선언했다. 이는 00000001000000010000000100000001 이다.

Result는 총 bit 숫자를 count하는 함수이다. Result=(a & x) + (a & (x >> 1)) + (a & (x >> 2)) + (a & (x >> 3)) + (a & (x >> 4)) + (a & (x >> 5)) + (a & (x >> 6)) + (a & (x >> 7))를 하면 1,9,17,25번째 비트를 통해 그 수로부터 앞의 7번째자리까지의 비트로 8개의 비트에서 1의 개수를 나타낼 수 있다. Result를 우측으로 16 shift시켜 기존의 결과값에 더하여 결과값을 갱신하면 앞의 32~16째자리 비트가 16~0번째 자리의 비트에 더해지고, 이를 우측으로 8번 더 shift시키고 결과값에 더하면 앞의 8비트에 1의 개수 정보가 저장된다. 따라서 result와 0xff값의 &값을 리턴하면 count된 1의 개수를 구할 수 있다.