20180910 마프

Unsigned Integer와 Two’s complement number의 multiplication을 생각해봄녀

Unsigned integer의 경우 0~(Umax)^2까지 범위를 가지게 된다(Umax=2^w-1)

Two’s complement number의 경우 Tmin\*Tmax~Tmax^2까지의 범위를 가지게 된다.

Power of 2 multiply( 2^n을 곱하는 연산)을 할 때는 그냥 왼쪽으로 n만큼 shift를 하면 된다. 이것은 signed나 unsigned나 아무 상관없음!!

Power of 2로 나누는 작업 (quotient of unsigned by power of 2)의 경우에도 그냥 오른쪽으로 shifting을 하면 된다. 그런데 이 경우 unsigned number에만 해당되며 2^n으로 나눈 다음 소수점은 버리는 가우스 연산을 생각해야한다. Unsigned number의 경우 arithmetic shift가 진행되지만(추가되는 곳에 0을 넣는것) signed number의 경우 logical shift가 적용되기 때문이다.(새로 추가되는 부분에 sign bit을 유지시키는 것)

Unsigned를 쓰는 이유! 이해하지 않고 쉽게 사용 가능하기 때문이다. Lab에서 경험해 봤겠지만 int를 사용할 경우 two’s complement를 이용하기 때문에 right shift가 sign bit를 보존하는 등 계산대로 잘 동작하지 않았다. 따라서 unsigned를 잘 이용하여 이런 문제를 해결해 주는 것이 좋다.

컴퓨터는 보통 virtual memory를 사용하는데 virtual memory의 경우 word size에 dependent하다.

Word size는 일반적으로 32,64bit 정도인데 32bit인 경우 0~2^32-1개의 virtual memory address가 존재할 것이다. 이게 확실하진 않은데 32bit라는 건 주소 하나마다 8bit이므로 4byte가 1word라는 뜻이며 64bit의 경우 주소 하나마다 8bit이므로 8byte가 1word라는 뜻인 것 같다. 즉 주소 하나하나마다 1byte인데 1byte에는 8bit씩 store가 되어있는 것! 그렇다면 64 bit에서도 virtual memory address는 2^32-1인가??

그리고 이런 virtual memory는 physical memory에 mapping된다.

이런 virtual memory의 경우 각각의 program마다 개별적으로 존재한다. 이 virtual memory의 경우 사실 실제로 physical한 공간이 아니기 때문에 이런식으로 사용이 가능한 것 같다.

Big Endian의 경우 우리가 보통 사용하는 방식으로 정보를 저장하지만 little endian의 경우 integer를 반대로 저장한다. 그런데 pdf에서처럼 한 박스 안에 숫자가 두개씩 들어가는 이유가 무엇일까..?? 저거 한 개가 16bit인데 1 word size가 32bit이기 때문인 것 같다!!

그런데 character representation의 경우 endian system이 무엇이든 신경안쓴다는거같기도하고… 헷갈림

Floating point representation같은 경우 2진수로 나타낼수 있긴 하지만 굉장히 작은 단위나 굉장히 큰 수의 경우 나타내기 곤란하다는 단점이 있었다 따라서 IEEE에서 standard를 만들었는데 이것이 IEEE Standard 754인듯하다. 이런 floating point representation standard의 경우 sign bit 1bit를 사용하고 exponential을 나타내는 exp bit, 그리고 실제로 1.0~2.0사이의 유효숫자를 나타내는 frac bit이 있다. Float number가 32bit인 경우 1+exp+frac=32 bit이 되어야 할 것이다. 그런데 exp와 frac에 들어있는 값이 정확하게 exp값을 나타내지 않고 frac에 들어있는 값이 정확히 유효숫자 그 자체가 아니다. 이것은 전부 Normalized value를 이용하여 encoding 되어있다.

이 경우 32bit는 single precision, 64bit의 경우 double precision이라고 하는데 Single의 경우 1+8+23, double의 경우 1+11+52bit으로 나타내는 것이 standard인 것 같고 extended precision이라고 해서 80bit짜리가 있긴 한데 흔치않은 케이스로 일반적이지 않은것같다.

Normalized value를 살펴보자 우리는 exp값을 계산할 때 일단 exp bit에 저장되어 있는 값에 –bias를 해 준다. 잘 생각해보면 bias를 2^(k-1)-1로 해주면(k=exp bit) E=Exp-Bias로 했을 때 원래 exp에 저장되는 값이 1~254라고 하면 (k=8일 때) Bias가 127이므로 E의 범위가 -126~127로 바뀌어서 많은 범위를 커버할수 있게 된다. 한마디로 unsigned 값이 저장되는 것을 signed가 표현될수 있도록 sign bit을 만들어주는 느낌이다.

Frac의 경우에도 1.xxxxxx로 나타나기 때문에 MSB인 1은 저장하지 않고 xxxxx부분만 잘 저장해준다. 이러면 1bit를 아낄수있음!!