2020314597 김주민

Design and Code Explanation

hpfp design

hpfp는 msb부터 1비트의 s비트, 5비트의 exp비트, 10비트의 frac비트를 가진 총 16비트의 자료형이다.

hpfp의 special value들과 그 비트 형태는 다음과 같다.
+∞ : 0 11111 0000000000
-∞ : 1 11111 0000000000
NaN : 0 11111 xxxxxxxxxx or 1 11111 xxxxxxxxxx (x = 0 or 1, frac != 0)

그러므로 hpfp가 나타낼 수 있는 최댓값과 최솟값은 다음과 같다.
+32736 : 0 11110 1111111111
-32736 : 1 11110 1111111111

code explanation

1. int to hpfp

- input에 대하여 hpfp 자료형의 범위를 넘어서는 값들과 0을 먼저 예외적으로 처리해준다.
- input이 양수인지 음수인지 확인해 s값을 정한다.
- input을 이진수로 표현했을 때의 MSB를 찾는다.
- input을 2씩 나누어가며 이진수표현의 reversed버전을 배열에 저장한다.
- mantissa에서 맨 앞 1을 제외한 값을 frac에 저장한다.
- frac이 10비트를 차지할 수 있게 left shift한다.
- exp를 구하고 left shift한다.
- s + exp + frac을 return한다.

2. hpfp to int

- input에 대하여 비트 패턴이 +∞나 -∞를 나타내면 각각 int의 최댓값, int의 최솟값을 return한다.
- 이 때 return 되지 않았는데 input 중 exp부분이 11111이면 NaN이므로 int의 최솟값을 return한다.
- frac에 bit shift를 한 뒤 맨 앞에 있다고 친 1을 실제로 붙여준다.
- input의 MSB비트의 부호에 따라 frac or -frac을 return한다.

3. float to hpfp

- input에 대하여 hpfp 자료형의 범위를 넘어서는 값들과 0을 먼저 예외적으로 처리해준다.
- float는 32비트고 hpfp는 16비트여서 비트 shift를 해야하는데 float와 double 자료형은 비트 연산이 안 된다고 한다. 그래서 Union을 써서 int처럼 사용했다.
- s, exp, frac은 비트 shift 과정에서 logical shift가 되길 원하기 때문에 unsigned int로 선언했다.
- s를 구하고 exp를 구한다.
- 헷갈릴 거 같아서 exp -127을 해 e에 따로 저장한다.

- 비트 shift를 통해 frac을 구한다.
- e가 -14보다 크면 normalized case가 확실하다. e + 15를 해 exp를 구한다.
- e가 -14면 noralized일 수도 있고, denormalized일 수도 있다. 그러므로 input값의 크기를 기준으로 나눠 exp를 구한다.
- e가 -14보다 작으면 hpfp로 표현하기엔 0과 너무 가까운 수이다. 0을 return한다.
- exp를 비트 shift한다.
- s + exp + frac을 return한다.

4. hpfp to float

- s와 frac은 비트 위치만 다르므로 비트 shift를 통해 위치를 잡는다.
- exp는 위치도 다르지만 Bias가 다르므로 float에 맞게 exp값을 변경해준 뒤 비트 shift를 한다.
- s + exp + frac을 return한다.

5. addition

- input a와 b에 대하여 a 혹은 b가 NaN일 때, a가 +∞일 때, a가 -∞일 때, b가 +∞일 때, b가 -∞일 때로 경우를 나누어 예외들을 처리한다.
- 부동소수점의 덧셈은 우선 지수 부분을 일치시켜야하기 때문에 더 큰 지수를 찾아서 더 작은 지수를 맞춰주었다. 그 만큼 frac을 shift 해줬다.
- frac들을 더한다. 근데 말이 더하는 거지 a와 b부호 다르면 빼줘야하기 때문에 더하는 거랑 빼는 거를 경우를 나누어서 수행했다.
- normalized case끼리 더하면 무조건 비트가 한 자리 늘어난다. 그럼 한 번 right shift해줘야 하는데 round to even 후 shift해준다.
- denormalized case일 때는 비트를 left shift해줘야할 수 있다. 그러면 left shift 후 그 만큼 exp에서 빼서 반영해준다.
- 더 큰 값의 부호로 return값의 부호를 정해준다.
- a와 b가 hpfp로 표현할 수 있는 수이지만 더했을 때 +∞이거나 -∞인 경우를 예외 처리해준다.
- s + exp + frac을 return한다.

6. multiply

- input a와 b에 대하여 a 혹은 b가 NaN일 때, a가 +∞일 때, a가 -∞일 때, b가 +∞일 때, b가 -∞일 때로 경우를 나누어 예외들을 처리한다.
- 부동소수점의 곱셈은 지수 부분을 더해야하는데 exp 상태로 더하면 Bias가 두 번 더해지는 꼴이기 때문에 Bias를 한 번 빼준다.
- frac들을 곱한다.
- frac의 곱은 무조건 소수점 아래 20자리가 생기고 소수점 위로는 1 혹은 1x가 생길 수 있으므로 둘의 경우를 나눈다.
- 각각의 경우에 대하여 round to even을 시행하고 frac이 10자리가 될 수 있도록 right shift한다.
맨 앞의 implied 1을 제외한 값을 저장한다.
- a, b를 통해 각각의 s를 구한다.

- a와 b가 hpfp로 표현할 수 있는 수이지만 곱했을 때 +∞이거나 -∞인 경우를 예외 처리해준다.
- s + exp + frac을 return한다.

7. comparision

- input a와 b에 대하여 a 혹은 b가 NaN일 때의 예외를 먼저 처리한다.
- s비교, exp비교, frac비교 순으로 비교를 진행하여 ">" or "<"를 return한다.
- 모든 경우에 비교값이 return이 안 되면 "="를 return한다.

8. hpfp to bits

- input값의 비트를 MSB부터 1과 마스킹하여 배열에 앞에서부터 채워넣는다.
- 처음에는 little endian으로 저장된 값을 불러왔으므로 0~7, 8~15번째 인덱스를 나누어서 계산하였다. 근데 저장할 때만 그렇게 되지 불러오면 또 바이트별로 나누어져서 저장된 값을 합쳐서 불러와지기 때문에 필요가 없었다.