16비트 float 자료형인 sfp 를 구현해서 조작하기 위해서는 비트 단위 연산을 하거나, sfp의 비트를 배열에 담아 배열로써 조작하는 두가지 방법이 있다. 후자가 좀 더 구현하기 편함으로, 이를 위한 bits2sfp, makesfp, bits2int 등의 함수를 추가로 정의하고 구현했다. Bits2sfp는 char형 배열을 받아 이를 sfp로 변환해주는 함수이며, bits2int도 이와 같다. Makesfp는 sign, exp, frac을 각기 파라미터로 받으며, 이를 함해 sfp로 변환하는 함수이다.

1. Int2sfp는 우선 exp를 알아내기 위해 임시변수 tmp를 선언하고 tmp가 0이될때까지 while 루프를 돌리면서 매번 exp를 증가시켰다. Frac을 측정하기 위해 input 값을 frac에 넣은 후 상위 9비트만을 측정하기 위해 exp-9만큼의 값만큼 오른쪽으로 쉬프트 해 주고 sfp2bits함수를 호출해 배열로 변환했다. 그 후 제일 왼쪽의 1이 나올 때 까지 i를 증가시키면서 측정한 후 그 다음 비트부터 makesfp의 인자로 전달했다.

2. Sfp2int는 주어진 sfp를 sfp2bits에 넣어 bits로 받은 뒤 exp=E+bias인 것을 통해 실제 제곱값인 E를 구하고 (-1)^sign*2^E*frac=input를 계산하기 위해 frac, sign 값을 구하고 위 식에 대입하여 input값을 계산하였다.

3. Float2sfp 함수는 float 자료형의 값 int형처럼 취급해 계산해서 frac과 exp, sign을 각각 구한 후 그 값을 적절한 변형을 통해 makesfp 함수에 인자로 전달하였다.

4. Sfp2float 함수 역시 bits2sfp함수로 sfp의 비트스트링을 구한 후 sign, exp, frac을 구해서 float자료형을 int인것처럼 취급해서 or 연산으로 넣어주었다.

5. Sfp_add 함수가 이번 과제에서 가장 구현하기 어려웠던 함수이다. 가장 우선적으로 infinity, NaN 체크를 한 뒤 정상적인 수+수 일 때 함수를 진행시킨다. 우선 input 2개를 각각 a,b로 설정하고 exp_a, exp_b를 구한 후 대소비교를 한다. 결국 표현될 수 있는 frac은 9bits 뿐임으로 exp가 작은 값의 frac을 오른쪽으로 두 exp의 차이만큼 쉬프트 해 준다. (ex, exp_a=33, exp_b=32일 경우 frac_b >>=1) 그 후 각 frac을 sign을 참고해서 더해준다. 만약 어떤 수가 음수라면 frac도 2의 보수를 취한 후 더한다. 그 후 가장 오른쪽 첫번째 비트와 두번째 비트를 참고해서 round-to-even 방식으로 올림/내림을 결정한다. 이렇게 구해진 frac을 bits라는 char형 배열에 각 비트별로 대입하고 가장 왼쪽 1을 제외한 체 makesfp 함수에 전달한다. 이때 exp는 둘 중 큰 값이며 frac에서 더해진 값이 exp보다 많은 자리수를 가지고 있을 경우 최대 +2까지 될 수 있다. Sign은 더해진 frac이 음수인지 아닌지를 확인해서 결정한다.