EE4U Project 4: Half Precision IEEE 754 Numbers

Student Name: Yi Li

1. Introduction

This project focuses on implementing encoding and decoding functions for half-precision IEEE 754 floating-point numbers according to the IEEE 754-2008 standard. Half-precision numbers are widely used in applications, especially in GPUs, as they use only half the memory compared to single-precision numbers.

2. Implementation Details

2.1 Half-Precision IEEE 754 Format

The half-precision format consists of:

- Sign bit: 1 bit
- Exponent: 5 bits
- Fraction: 10 bits
- Bias: 15 (2^(5-1) - 1)

2.2 Encoding Function Implementation

```c
void mp_hp_ieee754_encoding(float f, hp_IEEE754Field_TypeDef *pField)
```

The encoding function handles three main cases:

1. **Sign Bit Processing**:
   - Sets sign bit to 1 for negative numbers, 0 for positive numbers

- Works with absolute value for further processing

2. **Special Cases**:
   - Numbers too large (>= 2^16 - 2^5):
     - Sets exponent to 0x1F (all 1's)
     - Sets fraction to 0
     - Represents infinity

   - Denormalized numbers (< 2^-14):
     - Sets exponent to 0
     - Scales fraction to represent tiny numbers
     - Uses formula: fraction * 2^(-(bias + n_frac - 1))

3. **Normal Numbers**:
   - Uses `frexpf` to extract fraction and exponent
   - Adjusts exponent for implicit 1 in IEEE 754
   - Converts to half-precision format with proper bias

2.3 Decoding Function Implementation

```c
float mp_hp_ieee754_decoding(hp_IEEE754Field_TypeDef field)
```

The decoding function handles three cases:

1. **Infinity**:
   - When exponent = 0x1F
   - Returns INFINITY

2. **Denormalized Numbers**:
   - When exponent = 0
   - Uses formula: fraction * 2^(-(bias + n_frac - 1))

3. **Normal Numbers**:
   - Reconstructs fraction with implicit 1
   - Applies exponent with bias
   - Formula: (1 + fraction * 2^(-n_frac)) * 2^(exponent - bias)

3. Test Results

The implementation successfully passes all test cases:

1. **Encoding Tests**:
   - Too big number (2^18) -> Infinity representation
   - Normal number (-31.875) -> Correct sign, exponent, and fraction
   - Tiny number (2^-18) -> Proper denormalized representation

2. **Decoding Tests**:
   - Normal number (-31.875) -> Correct reconstruction
   - Tiny number (2^-18) -> Accurate denormalized number decoding

4. Key Features

- Compliant with IEEE 754-2008 standard
- Handles all special cases (infinity, denormalized numbers)
- Maintains precision within half-precision limits

- Efficient implementation using standard C math functions

5. Conclusion

The implementation successfully provides encoding and decoding functionality for half-precision IEEE 754 numbers, meeting all project requirements and passing all test cases. The code is efficient, handles edge cases appropriately, and maintains precision within the constraints of the half-precision format.

6. References

- IEEE 754-2008 Standard
- Project Documentation
- C Standard Library Documentation

7. Test Results Screenshots